TECHNICAL DOCUMENT 3051
December 1998

Wide-Area, Heterogeneous, Distributed Computing: Toward Computational Grids

C. V. Tran

Systems Center
San Diego

SSC San Diego
San Diego, CA 92152–5001

SSC San Diego
San Diego, California 92152–5001

H. A. Williams, CAPT, USN
Commanding Officer

R. C. Kolb
Executive Director

ADMINISTRATIVE INFORMATION

The work described in this Technical Document was performed by the Joint and National Systems Division, Code D73, SSC San Diego. Funding was provided under program element number 0603011F.

Released by
D. Matsubara, Head
National Sensors

Under authority of
T. A. Knight, Head
Joint and National
Systems Division

SB

CONTENTS

Figures

1. OVERVIEW

Since the late 1970s, many applications (including mission-critical ones) were recognized as suitable for distributed computing. The benefits of such data processing systems include high performance, high availability, high reliability, resource sharing, high adaptability to workload changes, and modular, incremental growth.

Some applications require distributed computing because of their inherent nature. For mission-critical applications where high availability (achieved through fault tolerance) is the foremost concern, redundancy of hardware, software, and data is required; and preferably, these resources are widely distributed.

To the user, a distributed computing system behaves like a "virtual uni-processor," which was how such a system was described in the 1970s. With the steady advent of communication infrastructure in the last two decades, a metacomputer, currently called a virtual shared-computing machine, can compute across a wide area, and even across continent(s). This "virtual supercomputer" provides a unified system image, making all computing resources transparently available to users.

A computational grid is a recently emerging concept analogous to an electrical power grid, offering potential access to computational resources in a ubiquitous, inexpensive, and dependable manner. This fresh point-of-view emphasizes the vastness of the future computing environment, and the need for infrastructure as well as the far-reaching consequences.

Today, computational grids do not exist. However, there has been extensive research over the last 5 years in wide-area, heterogeneous, distributed computing thanks to technology advances in microprocessors and interconnection networking. Widespread experimentation of large-scale distributed-computing testbeds has been initiated and operated at government and academic research laboratories. The main thrusts are to design, implement, and deploy various building blocks for the grid environment, and to explore enhanced capabilities of existing applications and new capabilities of promising grid-enabled applications.

This document provides an overview of recent developments in several important aspects of wide-area, heterogeneous, distributed computing. Section 2 presents information on affordable, high-performance, commodity computing clusters, a building block of computational grids. The rapid advent of mass-market, off-the-shelf microprocessors and high-speed networking has brought commodity cluster computing into focus as an affordable high-performance computing alternative for testbed development and deployment. Commodity clusters delivered scalable, high and sustained performance at an affordable cost at the Supercomputing Conference SC'97 in San Jose, CA, by co-winning a Gordon Bell Prize for price/performance. A real-world, large-scale example is the Whitney project of the Numerical Aerospace Simulation (NAS) Systems Division at NASA Ames Research Center. This project is integrating off-the-shelf hardware and software technologies to build a cluster of hundreds to thousands of nodes supporting scientific workload. These developments illustrate the potential of commodity clusters as a building block of computational grids. The lack of affordable, scalable, interface4s for network interconnects is one major challenge to commodity clusters. The industry-standard Virtual Interface Architecture (VIA) spearheaded by Intel, Compaq, and Microsoft, with contributions of over 100 industry and research organizations, recently addressed this problem. VIA will be immediately applied in the recently ratified IEEE-standard Gigabit Ethernet technology. The coupling of VIA and the emerging Gigabit Ethernet may

become the dominant force in short-haul cluster networking and, in turn, may provide the path to low-cost, high-performance, scalable commodity clusters.

Workload-management software (WMS) is another building block of computational grids. Essentially, WMS performs distributed, network-wide, job management by dynamically scheduling resources to virtually transform a network of heterogeneous platforms into a single, shared-computing machine. The four main requirements for WMS are network and resource awareness, load balancing, fault tolerance, and security.

Section 3 discusses the current state of WMS by identifying leading WMS packages and describes the most desirable features in detail. These packages have been used in massively parallel systems as well as in loosely coupled clusters. However, the current incapability of these packages is well-recognized. Of immediate importance is the lack of WMS interoperability. In a computational grid environment, resources are heterogeneous, located across many sites, and under different administrative domains. Most likely, resources are also under different WMS packages. The lack of WMS interoperability is an immediate problem for computational grids.

There are two ways to achieve WMS interoperability. One is through standardization. The PSCHED initiative, a recent standards effort, has not been successful because principal vendors have showed little interest. The other way to arrive at interoperability is to devise a high-level resource-allocation manager to interface with local WMS packages. The Globus project adopted the latter approach to build a grid resource-management architecture. This development is described in section 4.

The computational grid environment also motivates and enables new programming models and services. The main topic of section 4 is agent-based programming models. Essentially, software agents (or simply, agents) are software components that reside in a computing environment and are tasked to perform certain services. While a stationary agent can only run on one machine, a mobile agent can roam from one machine to another. Mobile agents could provide better support for mobile computing. Mobile platforms–such as laptop, notebook, and palm-sized computers as well as hand-held personal digital assistants (PDAs)–are characterized by limited storage and processing capacity, limited-capacity software and runtime environment, a low-bandwidth, high-latency connection, and a long-period disconnection. Since mobile agents can migrate and access resources in the network, they do not need constant connection to home platforms. Upon completing their tasks, mobile agents send results back to the home platforms or temporarily store results on a docking mechanism in case of disconnection. This operational concept facilitates mobile computing, thus enlarging the need and capability of computational grids. Alongside a growing number of commercial applications, the role of mobile computing in C^4I is well-recognized, especially with the vision of small, mobile, adaptable, quick-response command structures of the future.

Agent-based programming has implications on WMS. WMS can be agent-based; that is, WMS functionality can be performed by system-level agents. An example is the NetSolve scheduling agent(s) that can provide access to network resources for numerical-computing execution in a grid environment. However, the scheduling functionality can be assumed by application-level agent(s) for the user's application. In the Application-Level Scheduler (AppLeS), the scheduling agent is a stationary agent that coordinates four subsystems to select resources, to generate resource-dependent schedule, to implement the best schedule, and to monitor and predict network and resource performance. Other examples are mobile-agent models that either let mobile agents autonomously seek resources and migrate, or use model-based mechanisms (for example, a microeconomic, market-

oriented approach) for resource control. In general, to achieve efficient end-to-end resource utilization requires coordination between system-level scheduling and application-level scheduling where an application-level scheduler can specify the application's performance requirements.

Since there are many applications running simultaneously in a grid environment, application-level agents need policy and mechanisms for cooperation or the environment is subject to chaotic and dangerous behavior as well as a waste of resources. This is a crucial issue when thousands or even millions of agents are deployed in a computational grid environment. For mobile agents, another challenge is security. In particular, while agent transfer is essentially a multi-hop operation, multi-hop authentication is unavailable. The mobility of mobile-agents increase the need to protect not only resources (for example, platforms and databases) against agents, but to also protect agents against resources that can be programmed to compromise or even terminate the agents. When deploying mobile-agent-enabled applications in a closed grid environment, inadvertent damages are likely to occur through buggy software. The impacts on system security and integrity, however, are not unlike those that result from malicious attacks.

Building computing infrastructure for future grids is an ongoing process. Obviously, this is a significant activity with far-reaching impact in shaping the 21st century. For military applications, its implications touch many facets of future capabilities and missions.

2. MASS-MARKET, COMMODITY OFF-THE-SHELF COMPUTING CLUSTERS

2.1 INTRODUCTION

The deployment, at Sandia National Laboratories, of the Intel ASCI-Red massively parallel platform of the DoE Accelerated Strategic Computing Initiative (ASCI) project has aroused general interest in high-performance systems built from commodity, commercial off-the-shelf hardware. The Intel ASCI-Red, the first teraflops (trillion floating-point operations per second) computer, was assembled with more than 9000 commodity 200-MHz Pentium Pro processors. Only the networking components of the system are proprietary and custom-built to guarantee high-performance internode communications. The recent drop in prices of fast Ethernet (100 Mbps) networking hardware allows clusters up to hundreds of PCs assembled using available off-the-shelf CPUs, either the 200-MHz Pentium Pro processors or the faster Pentium II processors, to function as a high-performance computing platform. These commodity clusters are affordable, usable, reliable, easy to build, maintain, and upgrade, and above all, show good price/performance. Furthermore, commodity clusters provide quick-to-set-up prototypes and testbeds. For these reasons, a growing number of academic and government computing and research laboratories are using commodity clusters. Their popularity is enhanced by the capability demonstrated at the Supercomputing Conference SC'97 (San Jose, CA, November 1997) that captured a prestigious Gordon Bell Prize for price/performance. While traditional commodity clusters adopt the Linux operating system, there is an emerging trend using Windows NT. In April 1998, the High Performance Virtual Machine (HPVM) project at the University of Illinois, Urbana-Champaign (UIUC), under the sponsorship of the Defense Advanced Research Projects Agency (DARPA), successfully demonstrated a Windows NT cluster of 256 (300-MHz Pentium II) processors using 128 dual-processor nodes.

In section 2, we describe the hardware and software of a typical commodity cluster, its performance, advantages, and disadvantages. A list of selected commodity-cluster projects, Linux-based and NT-based, is provided. We also discuss the major technical challenge to communication performance in cluster computing, namely, that except for large-volume data transfers, many other applications are unable to take advantage of the increased network bandwidth unless host protocol overheads are lowered. Additionally, we discuss the emerging Virtual Interface Architecture (VIA) cluster-network standard and its approaching impact on the role of commodity-cluster computing in high-performance parallel and distributed computing.

2.2 HARDWARE

Currently, the typical hardware of a commodity cluster is based on the Intel 200-MHz Pentium Pro processor because of its low price. However, there are clusters built with 300-MHz Pentium II processors (e.g., HPVM at UIUC and Aeneas at UC Irvine). Pentium II-based clusters are quickly becoming common because their price/performance becomes affordable.

A typical commodity cluster is usually configured with one node serving as the front-end and the rest as computing nodes. Each node typically has one CPU, 128-MB RAM, a hard disk (2.5 GB or more), a floppy drive, one SVGA adapter, and one Fast-Ethernet adapter. The front-end might have a larger capacity and/or more hard disks for storage purposes, if desired. A monitor and a keyboard are shared among all PCs using video/keyboard switch boxes. Note that the SVGA adapters on the com-

puting nodes are for diagnostic purpose only, and thus, do not require the many features that might be needed on the front-end for multimedia capability. Furthermore, the front-end might have extra Fast-Ethernet adapter(s) to link with the outside world.

The networking of a typical commodity cluster is Fast-Ethernet technology that has the peak data rate of 100 Mbps (mega bits per second). Fast-Ethernet hardware, such as network-interface cards (NICs), hubs, and switches, has become a mass-market, off-the-shelf commodity sold at reasonable prices because of the migration of standard (10 Mbps) Ethernet technology to the Fast-Ethernet one. Most vendors no longer support two separate product lines, one for standard-Ethernet adapters and the other for Fast-Ethernet adapters. To ease the hardware migration path, vendors are now offering 10/100 Ethernet adapters using autosensing technology to run at 10 Mbps or 100 Mbps according to the speed of the hub or switch port. As of September 1998, top-of-the-line 10/100 PCI Ethernet adapters are priced under $70 each.

Network topology for a typical commodity cluster is either switch-based for a small cluster (of 16 nodes, for example), or based on a combination of switches and hubs for a large cluster up to hundreds (and possibly thousands) of nodes. Clearly, a configuration such as a 2D Torus has better network performance; however, because each node needs more network-adapter cards, the routing gets too copious to be practical for a network of reasonable size.

An alternative to off-the-shelf, commodity, Fast-Ethernet technology is more expensive gigabit-per-second networking such as the proprietary Myrinet from Myricom (http://www.myri.com). Myrinet is a family of commercial products that originated from research projects funded by DARPA at the California Institute of Technology and the University of Southern California. Currently, the peak data rate of the Myrinet adapters and switches is 1.28 Gbps, or 2.56 Gbps in full duplex. Another possibility is the emerging Gigabit Ethernet technology; its standard over fiber was ratified by IEEE in June 1998. The standardization for Gigabit Ethernet over copper is expected to be completed by mid-1999. It is believed that as soon as standardized products are available, Gigabit Ethernet will quickly become the widely adopted technology for short-haul networking.

We end this section by noting that there is a nine-page tutorial that describes how to build a 16-node commodity cluster such as an earlier configuration used at at the Calfornia Institute of Technology Center of Advanced Computing Research (Lindheim, 1997). Figure 1, taken from this tutorial with the author's permission, depicts a system setup of a 16-node commodity cluster using a Fast-Ethernet switch.

2.3 SOFTWARE

Traditionally, starting with the very first commodity cluster (called a Beowulf) assembled at NASA Goddard Space Flight Center in summer 1994 using Intel DX2 processors at 50 MHz, commodity clusters have favored Linux to leverage the availability of compilers and utilities.

Linux is a Unix clone operating system used primarily for the Intel processors. Eventually, many GNU compilers and utilities were ported to Linux. Common compilers and tools include:

- GNU C, C++ and FORTRAN 77 compilers,
- X11 with development libraries, including windows managers, such as fvwm and its Windows 95-emulated version, fvwm95,
- Networking software, including NIC drivers,

- Message-passing libraries such as MPICH and PVM,
- Portable Batch System (PBS) for cluster-queuing system, and Extensible Argonne Scheduling sYstem (EASY) for scheduling.

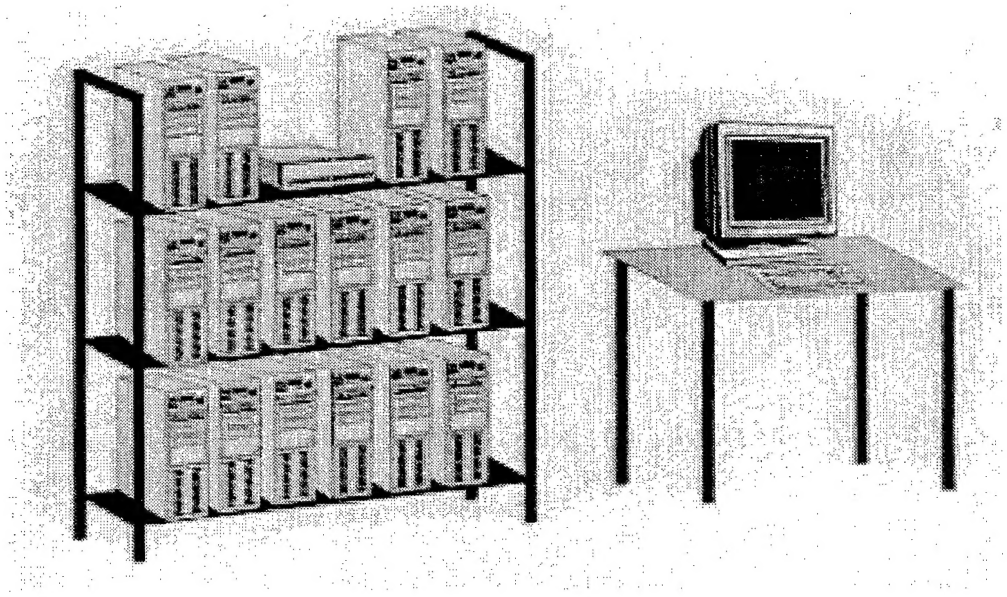

Figure 1. System setup of a 16-node commodity cluster using a Fast-Ethernet switch.

The Linux kernel and all of these compilers and utilities are free. Vendors package the Linux kernel, compilers, and various utilities together with their installation applications into so-called distribution and sell at a nominal price, usually under $50. Intended for stand-alone PCs, none of these distributions include message passing libraries (such as MPICH and PVM) and load-management software (such as PBS and EASY); however, these packages can be obtained directly from Internet sites. The most widely used distribution is Red Hat Linux from Red Hat Software, Inc. (http://www.redhat.com/). This distribution includes the Red Hat Package Manager (RPM) to facilitate software installing and upgrading. Red Hat Linux 5.1 was released on 1 June 1998. Version 5.2, due in November 1998, will support symmetric multiprocessing (SMP).

Some of the bundled compilers and utilities, however, are not high-quality products. For example, while GNU C and C++ compilers are respectable, the GNU FORTRAN 77 (g77) compiler is substandard. For this reason, many commodity-cluster sites choose to use PGF77, the Portland Group FORTRAN 77 compiler, instead (Fineburg and Pedretti, 1997). The Portland Group (http://www.pgroup.com) that built compilers for the Intel Paragon and the current Intel ASCI-Red computer, has ported to Linux its C, C++, FORTRAN 77, Fortran 90, and HPF (High Performance Fortran) compilers as well as the parallel-application profiler and debugger. These compilers and tools are optimized for the Intel Pentium Pro and, most recently, Pentium II processors.

Meanwhile, there are signs of the emergence of Windows NT-based commodity clusters because the most important building blocks are in place. These include:

- MPICH/NT, a port of the MPICH to Windows NT, currently available from University of Mississippi (Hebert and Skjellum, 1997),
- PVM support for Windows NT and Windows 95 (Geist, 1997),
- The leading load-management software LSF (Load Sharing Facility), from Platform Computing (http://www.platform.com), supports Windows NT,
- The Portland Group now offers compilers (C, F77, F90, and HPF) and performance profiler tool for Windows NT.

Several cluster-computing projects have been using Windows NT 4.0. Notably, the HPVM-III cluster at UIUC, consisting of 128 dual-processor nodes (using 300-MHz Pentium II processors) was successfully demonstrated in April 1998. HPVM-III has been using LSF for cluster managing.

Because of difficulties created by the small market share of Linux, it is strongly believed that commodity-cluster computing will have a far-reaching impact in high-performance parallel and distributed computing if it adopts a capable operating system with a major market share such as Windows NT.

2.4 PRICE/PERFORMANCE

The main goal of commodity clusters is not performance per se, but price/performance, that is, to get better performance while spending fewer dollars. So far, commodity clusters have achieved this goal. Two events support the argument:

- A Gordon Bell Prize for price/performance was awarded to a team for their results using an Intel 200-MHz Pentium Pro commodity cluster at the Supercomputing Conference SC'97 on 21 November 1997 (Karp, Lusk, and Bailey, 1998).
- The Whitney project at Numerical Aerospace Simulation (NAS) System Division, NASA Ames Research Center, Moffett Field, California, is integrating commodity off-the-shelf hardware and software technologies to incrementally build a cluster of hundreds to thousands of nodes to support scientific workload (Jones, 1998).

As described in the December 1997 issue of the *IEEE Computer* magazine, the Gordon Bell Prize was named after Gordon Bell who was a former National Science Foundation division director and is now a senior researcher at Microsoft Research. In his earlier career at Digital Equipment Corporation, Bell had been involved with the design of the PDP series. Since 1987, he has offered annual prizes to spur the transition of parallel processing from computer science research to useful applications. Entries to prizes are coordinated by the IEEE Computer Society.

The combined team consisting of California Insitute of Technology, Los Alamos National Laboratories, NASA Goddard Space Flight Center, and the University of Louvain, won the 1997 Gordon Bell Prize for price/performance using a commodity cluster. Indeed, California Institute of Technology and Los Alamos demonstrated their application at the Supercomputing Conference SC'96 in Pittsburgh. Their combined 32-node commodity cluster, each a 16-node 200-MHz Intel Pentium Pro cluster, performed a 10-million-particle, tree-code benchmark and achieved a 2.19 Gflops sustained performance. The cost of the combined system was $103,000 in 1996. The price/performance was 21 Gflops per million dollars. Unfortunately, the clusters and results were not available at the entry date, that is, 6 months before the conference. The quoted price/performance was three times better

than that of the price/performance winning team at the Supercomputing Conference SC'96 (Warren et al., 1997).

The Whitney project at the NAS Systems Division is equally interesting. It is named after Eli Whitney who is believed to be the first to use interchangeable parts in manufacturing. Although the notion of an affordable, off-the-shelf, commodity cluster was experimented with earlier at NASA Goddard Space Flight Center, the Whitney project is pushing it to a larger scale. The Whitney cluster is projected to have up to 500 processors by FY 99 and 5,000 processors in FY 2001 (Jones, 1998). The preliminary architectural design for booting, installing, and configuring nodes for a large cluster was conducted (Fineburg, 1997). Scalability and ease of maintenance was also considered. The project has performed benchmarking and performance analysis for network configuration (2D Torus, hub-based, switch-based, and hub/switch combination) on small clusters. The last configuration will be the hub/switch hybrid that considers performance, cost, and manageability (Fineburg and Preditti, 1997).

The Whitney project also performed a price/performance comparison for the current 200-MHz Pentium Pro clusters using the proprietary Myrinet versus the commodity off-the-shelf, Fast-Ethernet networking. The comparison was based on NAS Systems Division scientific workload exemplified by NAS Systems Division application benchmarks. The study concluded that although Myrinet had superior performance characteristics compared to Fast-Ethernet, at the current processor speed (200-MHz), it did not provide better price/performance until the cluster became extremely large (Becker, Nitzberg, and Wijngaart, 1997). Note that price/performance results are subject to revision as price changes. In fact, any drop in price implies better price/performance.

2.5 ADVANTAGES AND DISADVANTAGES

With the advent of mass-market, off-the-shelf commodity, high-performance microprocessors and networking, a low-cost, scaleable, fast computer can be assembled, and all the needed software can be installed in less than a week. The computer can achieve Gflops (billion floating-point operations per second) performance for $50,000. Commodity-cluster computing has recently been the current theme of high-performance computing in the environment of budget constraints and dissatisfaction with proprietary and expensive systems.

The many advantages of a commodity cluster can be summarized as follows:

- Affordability. This goal is achieved because all the hardware components are mass-market off-the-shelf. Software (commodity off-the-shelf operating system, utilities, compilers, message-passing libraries and other applications) is also free or at a nominal fee.
- Flexibility. A commodity cluster is highly flexible for upgrading and reconfiguration because of its modular construction.
- Ease of setting up testbeds. Again, this goal is achieved because of modular construction using standard off-the-shelf components.

While the affordability of commodity clusters is quite attractive in a budget-constrained environment, the flexibility for upgrading and reconfiguration allows quick response as the market evolves, and, thus, assures the investment's longevity. Evidently, in the long-run, a commodity cluster will be a heterogeneous system having a mix of several generations, each of a different type of performance and characteristics.

The major disadvantage of commodity clusters is believed to be the lack of a support facility for help when a problem arises (especially with the use of Linux as the operating system and the resulting software environment). This drawback, however, has been reduced by many online resources including Linux websites and newsgroups. In particular, the Linux Documentation Project is developing reliable documentation for installing, using, and running Linux.

Currently, the main disadvantage of commodity clusters is caused by the small market share of Linux. The freely available Linux operating system together with off-the-shelf, mass-market PCs and networking components has nurtured commodity clusters. Currently, the small market share of Linux platforms prevents commodity-computing clusters from playing a bigger role in parallel/ distributed computing.

The lack of vendor-supplied hardware drivers shows one disadvantage of the small market share of Linux platforms. All Linux sytem hardware device drivers are essentially written by Linux users. As a case in point, most vendors of Fast-Ethernet adapters supply corresponding drivers supporting Windows NT, Windows 95, Windows for Workgroup, Novell Netware, and several Unix flavors, but none support Linux. Most Linux Ethernet drivers were written by Donald Becker, a member of the team that won the 1997 Gordon Bell Prize for price/performance. The name, Becker Series Drivers, has been coined by the Linux community. Since the hardware components evolve rapidly, the smallmarket share of Linux platforms poses a serious drawback. Steve Elbert, at the Pentium Pro Cluster Computing Workshop, 10-11 April 1997, Des Moines, Iowa, related the following experience: a Fast-Ethernet card ordered a month later had different components and required a different driver update (Elbert, 1997). This market-driven disadvantage requires great capability on the part of users.

Additionally, the use of Linux as the operating system prevents new technology from spreading to commodity clusters, and, thus, holds back their capability and limits their usefulness. The situation is expected to improve quickly with Intel's recent decision to invest in Red Hat Software Inc. and to create a deeper involvement with the Linux community. However, the pervasiveness of Windows NT platforms offers a compelling reason for using Windows NT-based commodity clusters alongside Linux-based commodity clusters. This prescription is based on the reality that economics favors high-volume markets. As hardware becomes cheaper, software development becomes expensive and must rely on modular, plugged-in components that are only available on platforms that constitute reasonable market shares.

2.6 NETWORK-INTERFACE CHALLENGE AND INDUSTRY-STANDARDS EFFORT

Typically, a cluster interconnect network is a small-scale local-area network (LAN), often within an enclosed space, that connects cluster elements at high speeds. In current terminology, this is a system area network (SAN). Traditional LAN networking protocols were first developed for long-haul communications. End-to-end protocol overheads are substantially high, often several times the transport latency. So, the increase in bandwidth alone does not benefit applications that send and receive small data packets. Such applications include database operations and fine-grained parallel programs. Additionally, multiple data copies are required as messages pass from one cluster element to another. The corresponding overheads for copies and flushes are the dominant factor in communication performance (Keeton, Anderson, and Patterson, 1995).

Proprietary SAN interconnects have been provided by vendors. Two examples are Myrinet from Myricom and ServerNet from Tandem (now Compaq). Proprietary interconnects are expensive; such

networks can easily be more than half the total cost of the entire cluster (Sterling et al., 1998). However, academic and research organizations have been actively working on prototypes for SAN interfaces. Notably, U-Net (Cornell), Active Messages (UC Berkeley), Fast Messages (UIUC), and Virtual Memory-Mapped Communication (Princeton). These prototypes are efforts toward low-latency networking for SANs. The common features of these research-oriented network interfaces are (1) user-level protected network access, (2) hardware support for gather-scatter, and (3) early demultiplexing of incoming traffic. User-level network access reduces overheads by allowing data transfer without kernel involvement. For security reasons, this direct access must be in a protected mode. Hardware support for gather-scatter and early demultiplexing help eliminate message copies (Chien, 1998a). The benefits of these efforts, however, have not been transferred to commercial products.

Realizing that competing technologies tend to confuse rather than promote commodity-cluster computing, Intel, in collaboration with Compaq and Microsoft, initiated a standardization effort. In December 1997, Intel, Compaq, and Microsoft, with contributions from over 100 industry and research organizations, released version 1.0 of the Virtual Interface Architecture (VIA) specification. The three main features for high-performance network interfaces formed the backbone of the specification. User-level protected access allows the application to bypass the operating system and use its virtual memory to communicate directly to the network-interface cards (NICs). Standard interfaces for gather-scatter and early demultiplexing further promote zero-copy operations by eliminating buffer copying and kernel overhead. In short, with the VIA, control and setup go through the kernel, however, data go directly from (or to) the application to (or from) the NIC (Compaq, Intel, Microsoft, 1997; Compaq, 1998; Intel, 1997; Chien, 1998a).

VIA is intended to be operating-system independent and processor-independent. However, it stops short of providing the concrete API required for software portability. This shortcoming is expected to be overcome as the standardization process progresses.

Taking the role of promoters, the three vendors (Intel, Compaq, and Microsoft) are spearheading development of various VIA-enabled hardware and software products. It is expected that the immediate target of VIA is the emerging Gigabit Ethernet technology.

Overall, the VIA cluster network standard is an attempt to formulate a standardized, cost-effective, high-performance cluster-interconnect solution. The widespread adoption of VIA, when realized, will have a significant impact on the role of commodity-cluster computing in high-performance parallel and distributed computing in the near-term. In particular, VIA brings volume to high-performance cluster interconnects, thus lowering cost and promoting commodity off-the-shelf cluster computing. The coupling of VIA and the emerging Gigabit Ethernet technology will be the dominant driving force in SANs. This, in turn, will provide the path to low-cost, high-performance, scalable, commodity-cluster computing.

2.7 COMMODITY CLUSTER PROJECTS

While the majority of commodity clusters is Linux-based, there is an emergence of Windows NT clusters. Selected clusters of both groups are listed below.

2.7.1 Linux-based Commodity Clusters

The following list is incomplete; we list only four commodity clusters that have been referred to earlier in the section. The cluster, Aeneas, at the Department of Physics, Univeristy of California,

Irvine, is an example of a commodity cluster built on the 300-MHz Pentium II processors. Naegling, at the California Insitute of Technology, is the outgrowth of the original cluster (Hyglac), that together with Loki, of Los Alamos National Laboratories, won the 1997 Gordon Bell Prize for price/performance. Finally, Whitney, of the NAS Systems Division, NASA Ames Research Center, will eventually reach 5000 processors. Pointers to other Linux-based commodity clusters (and resources) can be found on home pages of the four mentioned clusters.

- Aeneas, Department of Physics, University of California, Irvine
  http://aeneas.ps.uci.edu/aeneas/
- This is a 65-node commodity cluster (1 front-end and 64 compute nodes), each node having a 300-MHz Pentium II processor, 128-MB RAM, a 3.1-GB EIDE disk, and one Fast-Ethernet card. The networking configuration is switch-based, using two 36-port Fast-Ethernet switches.
- Loki, Los Alamos National Laboratories
  http://loki-www.lanl.gov/
- Loki is a 16-node commodity cluster, each node having a 200-MHz Pentium Pro processor, 128-MB RAM, a 3.2-GB EIDE disk, and a Fast-Ethernet card. The network has a hypercube configuration.
- Naegling, Center for Advanced Computing Research (CACR), California Institute of Technology
  http://www.cacr.caltech.edu/research/beowulf/

This commodity cluster is currently composed of 114 nodes and two front-end systems, each having a 200-MHz Pentium Pro processor, 128-MB RAM, a 3.1-GB EIDE disk, and a Fast-Ethernet card. Each front-end system also has 128-MB extra RAM and an 8-GB extra disk. The network configuration is switch-based.

- Whitney, Numerical Aerospace Simulation (NAS) Systems Division, NASA Ames Research Center http://parallel.nas.nasa.gov/Parallel/Projects/Whitney/

According to an NAS Systems Division report, NAS-97-024 (Fineburg, 1997), as of October 1997, Whitney had 36 compute nodes, 3 I/O nodes, and 1 front-end. As of April 1998, two extra front-end systems were added. Each compute node has a 200-MHz Pentium Pro processor, 128-MB RAM, a 2.5-GB EIDE disk, and a Fast-Ethernet adapter. The I/O nodes are similar to compute nodes, except each has two 200-MHz Pentium Pro processors, 256-MB RAM, and two 9-GB SCCI disks. One front-end is a uniprocessor system with 512-MB RAM, a 4-GB SCCI disk, and a second Fast-Ethernet card for connecting to a general NAS network and the Internet. Two added front-ends are 333-MHz Pentium II systems with 128-MB and 512-MB RAM, respectively. The final network configuration is expected to be a switch/hub hybrid.

2.7.2 Windows NT-based Commodity Clusters

There is an emerging trend of commodity clusters shifting from Linux to Windows NT. Two important projects are:

- High Performance Virtual Machines (HPVM), University of Illinois, Urbana-Champaign.
  http://www-csag.cs.uiuc.edu/projects/hpvm.html

The HPVM III cluster consists of 128 dual-processor machines using 300-MHz Pentium II processors. It has in aggregate 48 GB of DRAM and 400 GB of disk space. It uses the Myrinet network

interface. The core communication layer is Fast Messages (FM), a low-level layer designed to enable convenient layering of other APIs and protocols on top of it (Chien, 1998b; Lauria, Pakin, and Chien, 1998). In particular, the MPI-FM is a high-performance implementation of the Message Passing Interface (MPI) based on the port of MPICH (the implementation of MPI by Argonne National Laboratory and University of Mississippi) to FM (Lauria and Chien, 1997).

- Virtual-Memory-Mapped Communication (VMMC) II/NT, Princeton
  http://www.cs.princeton.edu/shrimp/ntvmmc/

VMMC-II for Windows NT is a part of the Scalable High-Performance Really Inexpensive Multi-Processor (SHRIMP) project at Princeton University to investigate high-performance servers built from networks of commodity PCs and commodity operating systems. Currently, the VMMC-II/NT cluster has eight uniprocessor PCs, each with a 300-MHz Pentium II processor and 128-MB RAM. The VMMC model allows processes to transfer data directly between virtual memory buffers. VMMC-II/NT is an implementation of the VMMC model on the Myrinet network interface (Dubnicki, Cezary, Bilas, Chen, Damianakis, and Li, 1997; Dubnicki, Bilas, Li, and Philbin, 1997). Twenty-four more PCs will be added to the current cluster.

2.8 CONCLUSIONS

Microprocessor and network technology commodity clusters assembled from mass-market, off-the-shelf components can now deliver scalable, high and sustained performance at an affordable cost. The Linux operating system, instrumental in providing the first software environment for most commodity clusters, has reached the point where, because of the market force, it tends to hold back the capability and limit the usefulness of commodity clusters in parallel and distributed computing. The situation is expected to improve significantly because of Intel's recent decision to invest in Red Hat Software Inc. and to create a deeper technical involvement with the Linux community. However, the pervasiveness of Windows-NT platforms offers a compelling reason for using NT-based commodity clusters alongside Linux-based commodity clusters. In the near-term, the widespread adoption of the industry-standard VIA, when realized, will provide a standardized, cost-effective, high-performance, cluster-interconnect solution. The coupling of VIA and the emerging Gigabit Ethernet is expected to be the dominant driving force in cluster networking. This, in turn, will provide the path to low-cost, high-performance scalable clusters.

2.9 BIBLIOGRAPHY

1. The Beowulf Project at Center of Excellence in Space Data and Information Sciences (CESDIS), Goddard Space Flight Center (GSFC)

   http://cesdis.gsfc.nasa.gov/linix/beowulf/beowulf.html

The Beowulf Project was started at CESDIS in Summer 1994 with a 16-node (50-MHz Intel DX2) cluster. This project helped popularize mass-market, commodity, off-the-shelf computing clusters. Over time, the Beowulf Project at CESDIS contributed greatly to the advent of commodity-cluster computing. In particular, its contribution to Linux Ethernet device drivers is critical for the growth of (Linux-based) commodity clusters. The project's home page serves as a clearinghouse for information on Linux-based clusters.

2. The Berkeley Network of Workstations (NOW) Project

   http://now/CS.Berkeley.EDU/

The Berkeley NOW Project has been an innovative frontrunner in workstation-cluster computing. The project has assembled large (100-plus) Solaris-based UltraSPARC systems. It has also designed the fast communication interface Active Messages, now in version 2.0.

3. The Virtual Interface Architecture (VIA)

   http://www.viarch.org

The VIA specification (version 1.0, December 1997) can be downloaded after registration. Intel, the principal promoter of VIA, maintains a web page full of information for general viewers and for developers. The URL is http://www.intel.com/design/servers/vi/index.htm

Also of interest is the Berkeley VIA Project (http://www.cs.berkeley.edu/~philipb/via/) that is implementing the VIA for use in cluster networking for another project. The Berkeley VIA Project aims to develop high-quality VIA implementations for multiplatforms and to investigate possible improvements.

2.10 REFERENCES

Becker, J., B. Nitzberg, and R. Van der Wijngaart. 1997. *Predicting Cost/Performance Trade-off for Whitney: A Commodity Computing Cluster*. NAS Systems Division Technical Report NAS-97-023. NAS Systems Division, NASA Ames Research Center, Moffett Field, CA. October.

http://sciense.nas.nasa.gov/Pubs/TechReports/NASreports/NAS-97-023/

Available in FrameMaker and PostScript formats.

Chien, A. 1998a. "Computing Platforms." *The Grid: Blueprint for a New Computing Infrastructure.* Editors Ian Foster and Karl Kesselman, Morgan Kaufmann Publishers, San Francisco, CA.

Chien, A. 1998b. "High Performance Virtual Machines (Year 2 Status Report)."Viewgraphs presented at *DARPA ITO Quorum/HCC Meeting*, July 13-17, San Diego, CA.

http://www.dyncorp-is.com/darpa/meetings/quo98jul/presentations/CHIEN-PI981.ppt

Compaq Computer Corporation. 1998. *Virtual Interface Architecture for SANs*. Document Number ECG056/0698. June.

ftp://ftp.compaq.com/pub/suportinformation/papers/ecg0580698.pdf

Compaq Computer Corporation, Intel Corporation, and Microsoft Corporation. 1997. *Virtual Interface Architecture Specification*. Version 1.0. December 16.

Available to the public; free download upon registration.

http://www.viarch.org/

Dubnicki, C., A. Bilas, Y. Chen, S. Damianakis, and K. Li. 1997. "VMMC-2: Efficient Support for Reliable, Connection-Oriented Communication." Paper presented at *Hot Interconnects V Symposium*, August, Palo Alto, CA.

http://www.cs.princeton.edu/shrimp/Papers/hotIC97VMMC2.ps

Dubnicki, C,, A. Bilas, K. Li, and J. Philbin. 1997. "Design and Implementation of Virtual Memory-Mapped Communication on Myrinet." P*roceedings of the 11th International Parallel Processing Symposium (IPPS'97)* (pp. 388-396). April, Geneva, Switzerland.

http://www.cs.princeton.edu/shrimp/Papers/ipps97VMMC.ps

Elbert, S. 1997. "PCC Workshop Lessons Learned." Viewgraphs presented at the *Pentium Pro Cluster Workshop*, April 10-11, Des Moines, IA.

http://www.scl.ameslab.gov/workshops/Talks/Elbert/index.html

Fineburg, S. 1997. "A Scaleable Software Architecture Booting and Configuring Nodes in the Whitney Commodity Testbed." NAS Systems Division Technical Report NAS-97-024 (Oct). NAS Division, NASA Ames Research Center, Moffett Field, CA.

http://sciense.nas.nasa.gov/Pubs/TechReports/NASreports/NAS-97-024/

Available in PostScript format.

Fineburg, S. and K. Pedretti. 1997. "Analysis of 100Mb/s Ethernet for the Whitney Commodity Computing Testbed." NAS Systems Division Technical Report NAS-97-025 (Oct). NAS Systems Division, NASA Ames Research Center, Moffett Field, CA.

http://sciense.nas.nasa.gov/Pubs/TechReports/NASreports/NAS-97-025/

Available in PostScript format.

Geist, Al. 1997. "PVM on Pentium Clusters. Communication Spanning NT and UNIX." Viewgraphs presented at the *Pentium Pro Cluster Workshop*, April 10-11, Des Moines, IA.

http://www.scl.ameslab.gov/workshops/Talks/Geist/index.html

Hebert, S. and A. Skjellum. 1997. "MPI, Myrinet, and Pentium Pro Clustering." Viewgraphs presented at the *Pentium Pro Cluster Workshop*, April 10-11, Des Moines, IA.

http://www.scl.ameslab.gov/workshops/Talks/Skjellum/index.html

Intel Corporation. 1997. *Virtual Interface (VI) Architecture: Defining the Path to Low-Cost High-Performance Scalable Clusters.*

ftp://download.intel.com/design/servers/vi/scaleVI.pdf

Jones, J. P. 1998. "Overview of the Whitney Project: A Commodity Computing Cluster." Viewgraphs presented at *NASA Langley Research Center Parallel Systems SIG Meeting*. NAS Systems Division, NASA Ames Research Center, CA. February.

http://parallel.nas.nasa.gov/Parallel/Projects/Whitney/papers/LaRC-jjones.Feb98.ps

Karp, A, L. Ewing, and D. Bailey. 1998. "1997 Gordon Bell Prize Winners," *Computer*, vol. 31, no. 1 (Jan), pp. 86-92.

Keeton, K., T. Anderson, and D. Patterson. 1995. "LogP Quantified: The Case for Low-Overhead Local Area Networks." Paper presented at *The Hot Interconnects III Symposium*, August, Palo Alto, CA.

http://now.CS.Berkeley.EDU/Papers/Papers/hotinter95-tcp.ps

Lauria, Mario, Scott Pakin, and Andrew Chien. 1998. "Efficient Layering for High Speed Communication: Fast Messages 2.x" *Proceedings of the 7th High-Performance Distributed Computing (HPDC7),* July, Chicago, IL.

http://www-csag.cs.uiuc.edu/papers/hpdc7-lauria.ps

Lauria, M. and A. Chien. 1997. "MPI-FM: High Performance MPI on Workstation Cluster," *Journal of Parallel and Distributed Computing* (Jan), vol. 40, no. 1, pp. 4-18.

http://www-csag.cs.uiuc.edu/papers/jpdc97-normal.ps

Lindheim, J. 1997. *Building a Beowulf System.* CACR, California Institute of Technology, Oct. 10.

http://www.cacr.caltech.edu/research/beowulf/tutorial/

Available in HTML, PostScript, and PDF formats.

Sterling, T., T. Cwik, D. Becker, J. Salmon, M. Warren, and Bill Nitzberg. 1998. "An Assessment of Beowulf-Class Computing for NASA Requirements: Initial Findings from the First NASA Workshop on Beowulf-Class Clustered Computing." *Proceedings of IEEE Aerospace Conference*, March, Aspen, CO.

http://loki-www.lanl.gov/papers/ieee_aero98/p321.ps

Warren, M., J. Salmon, D. Becker, M. Goda, T. Sterling, and G. Winckelmans. 1997. "Pentium Pro Inside: I. A Tree Code at 430 Gigaflops on ASCI Red; II. Price/Performance of $50/Mflop on Loki and Hyglac." *Proceedings of Supercomputing'97 (SC'97)*, November, San Jose, CA.

http://loki-www.lanl.gov/papers/sc95/sc95.ps

3. WORKLOAD-MANAGEMENT SOFTWARE

3.1 INTRODUCTION

The proliferation of clusters of commodity PCs, workstations, parallel systems, and even clusters of clusters under the notion of distributed metacomputing has brought workload-management software (WMS) into focus. The distributed metacomputing concept encompasses a collection of platforms that virtually operate as a single, shared-computing resource. WMS is expected to perform network-wide workload management by dynamically scheduling resources and transforming a distributed network of heterogeneous platforms into a metacomputer.

The purpose of this section is to answer two questions:

- What are the leading WMS packages?
- What features are desirable when choosing a WMS package for a generic distributed computing environment?

A series of evaluations has recently answered the first question. Authors have assessed packages using documents (such as manuals and user's guides) available on the Internet with feedback from vendors/developers (in some cases). The evaluation results give snapshots of the state of WMS, though they may no longer be valid for the current releases. Moreover, these evaluations are essentially paper exercises without physically setting up, installing, and testing packages on different platforms. In addition, even if testing is performed, many issues are rather subjective and difficult to quantify (Baker, Fox, and Yau, 1995). For these reasons, it is imperative to distill a list of features that are most desirable for a generic distributed computing environment. This list, in turn, provides a general framework when selecting WMS. The references at the end of section 3 provide details for specific requirements.

3.2 REVIEW OF RECENT EVALUATIONS

In this subsection, we present recent evaluation results of WMS packages. These results, from different assessments, appeared in:

- A report for the British Joint Information Systems Committee (JISC) New Technology Sub Committee (NTSC) in November 1995. This report is also available as an online article in the DARPA-funded National High-Performance Software Exchange (NHSE) Review (1996 Volume, First Issue). It also appeared as a technical report of the Northeast Parallel Architectures Center (NPAC) at Syracuse University (Baker, Fox, and Yau, 1995).
- A White Paper of the Distributed Object Computational Testbed (DOCT) at the San Diego Supercomputer Center (SDSC). This document was an evaluation of WMS for use in DOCT (San Diego Supercomputer Center, 1997).
- A series of reports from the Numerical Aerospace Simulation (NAS) Systems Division, NASA Ames Research Center, on WMS in conjunction with Affordable High-Performance Computing, a NASA cooperative project (Jones, 1996a; Jones, 1996b; Jones and Brickell, 1997).

The NPAC report (Baker, Fox, and Yau, 1995) provided an extensive update of an earlier WMS evaluation conducted by NASA Langley Research Center (LaRC) (Kaplan and Nelson, 1994). Essentially using the LaRC's set of criteria, the NPAC report produced a short list of four commercial WMS packages (out of seven) and two research packages (out of 12) that met the evaluation standards in the 1995 period. The four leading commercial WMS packages, in alphabetical order, were:

- CODINE (COmputing in DIstributed NEtwork)
  http://www.genias.de/products/codine
- CONNECT:queue
  http://www.sterling.com
- LSF (Load Sharing Facility)
  http://www.platform.com
- NQE (Network Queuing Environment)
  http://www.cray.com/products/software/nqe/

The two research-oriented WMS packages on the short list, in alphabetical order, were:

- DQS (Distributed Queueing System)
  http://www.scri.fsu.edu/~pasko/dqs.html
- PBS (Portable Batch System)
  http://science.nas.nasa.gov/Software/PBS/

Historically, the Network Queuing System (NQS), developed at NAS Systems Division, is considered the progenitor of all WMS packages (Kaplan and Nelson, 1994). Three direct descendants of NQS are CONNECT:queue, NQE, and PBS. CONNECT:queue, a commercial product that originated from the software developed for the NAS Systems Division by Sterling Software, is no longer available. Network Queuing Environment (NQE) was originally developed to support the Cray Research Inc. (CRI) product line. Since the merging of CRI into Silicon Graphics, NQE has been ported to SGI platforms. Portable Batch System (PBS) has been developed and deployed at the NAS Systems Division for use on parallel and distributed systems.

The research-oriented Distributed Queueing System (DQS), developed at Florida State University, has a commercial version labeled CODINE (COmputing in DIstributed NEtwork) that is marketed by GENIAS Software GmbH (Germany), mostly in Europe. Load Sharing Facility (LSF), a product of Platform Computing, has its root in the research software, Utopia, developed at the Univeristy of California, Berkeley, and the University of Toronto, Canada.

Currently, LSF and CODINE are the two most widely used WMS packages, LSF in North America and CODINE in Europe. Platform Computing has successfully allied itself with major system vendors such as Sun, HP, Compaq, Silicon Graphics, Digital, Fujitsu, Hitachi, NEC, and Sony, who bundle, distribute, and co-market LSF to their customers.

A more recent evaluation of WMS was conducted for use in the DARPA-funded Distributed Object Computation Testbed (DOCT) at the San Diego Supercomputer Center. As noted in the online white paper dated January 1997, the evaluation concentrated on commercial packages, realizing that a commercial off-the-shelf product was more suitable for the project. The four leading WMS packages that appeared in the DOCT's short list, in alphabetical order, were:

- CODINE (COmputing in DIstributed NEtwork)
  http://www.genias.de/products/codine
- LoadLeveler
  http://www.rs6000.ibm.com/software/sp_products/loadlev.html
- LSF (Load Sharing Facility)
  http://www.platform.com
- NQE (Network Queuing Environment)
  http://www.cray.com/products/software/nqe/

This list is quite consistent with the commercial short list produced by NPAC, noting that CONNECT:queue is no longer available. LoadLeveler was not in the NPAC commercial short list because it only supported IBM SP-2 in 1995 when the NPAC evaluation was conducted. Since then, LoadLeveler has also supported heterogeneous platforms.

In the final analysis, the DOCT project chose NQE because (1) it allowed the insertion of an external scheduler, and (2) project members were most familiar with it. At the time of evaluation, NQE, LoadLeveler, and to some extent, CODINE, were commercial packages with interfaces for an external scheduler. Starting with version 3.1 (released in December 1997), LSF also provided this facility.

The most recent evaluation of WMS packages was performed with the NASA Affordable High Performance Computing (AHPC) project (Jones and Brickell, 1997). This evaluation is the second attempt of Phase 1 of a multiphase effort. Phase 1 was tasked to perform a pencil-and-paper comparison of WMS packages against stated capabilities based on NAS System Division requirements and vendor-supplied document. In Phase 2, the NAS System Division staff and selected users for each package will test in a less rigorous environment, meeting the minimum requirements used in Phase 1. Finally, in the optional Phase 3, testing will be done in a production environment with normal user workload for a 2-month evaluation (Jones, 1996b).

The original Phase 1 evaluation was performed in early 1996 at the NAS Systems Division. The following four leading WMS packages were ranked, highest to lowest: PBS, LSF, LoadLeveler, and NQE. However, because of incapability across the market, the Phase 2 evaluation was postponed and the Phase 1 evaluation was repeated in early 1997 (Jones, 1996b). The second Phase 1 evaluation gave the following list, from highest to lowest ranking: PBS, LSF, CODINE, LoadLeveler, DQS, and NQE. Again, Phase 2 evaluation was postponed because of incapability across WMS. Another Phase 1 evaluation is planned for a later date (Jones and Brickell, 1997).

The evaluation results of the leading WMS packages can be summarized in two short lists in alphabetical order:

Commercial: CODINE, LoadLeveler, LSF, and NQE.

Research: DQS and PBS.

In general, LoadLeveler, LSF, NQE, and PBS provide an exceptional service by making up-to-date, comprehensive documentation (including manuals, user's guides, administrator's guides, and white papers) available for online viewing and ready for download. This service is critical for a pencil-and-paper assessment.

As an update, we note that most packages have had upgraded versions with new features and improvements while others have been unchanged since mid-1997 when the NAS System Division report of the second Phase 1 evaluation was published. In particular, LoadLeveler has not changed; its current version is 1.3, released in August 1996. CODINE 4.2 is expected to be released in October 1998. Version 1.1.12 of PBS has been available since July 1998. NQE 3.3 was released in March 1998. LSF has been upgraded twice since the NAS Systems Division second Phase 1 evaluation; its current version is 3.2, released in August 1998. LSF has been the de facto leader in WMS. LSF is the WMS used to manage the SGI/Cray ASCI Blue Mountain at Los Alamos National Laboratory. Blue Mountain is a SGI/Cray Origin2000, a scalable shared-memory multiprocessor with 1024 processors. As noted in section 2, LSF is used for workload management in the 128-node (dual-processor) Pentium II cluster HPVM III at the University of Illinois, Urbana-Champaign.

There is an article on WMS from the user's perspective in the April-June (1998) issue of *IEEE Computational Science & Engineering* (Papakhian, 1998). Although the title indicates a comparison, the article essentially describes functionality and features of WMS packages. Overall, comparing WMS packages is a difficult exercise. It is subjective and many issues (including ease of use, configurability, maintainability, and user support) are hard to quantify (Baker, Fox, and Yau, 1995; Papakhian, 1998). For this reason, in the next section, a list of features most desirable for a generic distributed computing environment is presented to serve as a general guide for selecting WMS. The references at the end of this section provide details for specific requirements.

3.3 DESIRED FUNCTIONALITY

A typical WMS package allows enforcement of resource allocation limits using common parameters including

- number of CPUs per nodes
- number of nodes per job
- type of nodes per job
- number of jobs executing per user
- number of jobs executing per group
- wall clock time
- CPU time (per node and per application)
- system time
- memory utilization
- disk usage
- swap space

Essentially, WMS maps a job's resource requirement onto the available resource(s) in the metacomputing environment.

In addition to this basic functionality, many other features should be considered when choosing a WMS package. The 10 most-desired features are listed and discussed here.

a. Operate in a heterogeneous environment

A homogeneous computing environment consists of many platforms with the same architecture running the same operating system, while a heterogeneous environment is characterized by platforms of dissimilar architectures and different operating systems. Every environment will eventually become heterogeneous. Even an environment that starts from a homogeneous system will gradually become a mixture of several generations, each with different performance characteristics because of expansion, upgrading, and reconfiguration. Heterogeneity also occurs normally in wide-area distributed computing where the notion of the metacomputer encompasses computing resources at several sites, possibly geographically dispersed and under different administrative domains. Consequently, a WMS package that supports only a homogeneous environment is limited in distributed computing.

b. No single point of failure (SPF)

A single point of failure (SPF) occurs when the WMS package only runs with one designated master host. In such a case, when the master host fails, the whole system fails. For fault tolerance, some WMS packages allow the designation of an alternate master host. Preferably, every host other than the master host should serve as an alternate host so that when the master host fails, another host takes over, and the load-management services are available as long as any host is operational.

c. Provide security protection and support distributed file systems

Access control to remote execution must be provided. The two most common authentication approaches are: (1) weak authentication represented by an identification protocol such as RFC-931 (St. Johns, 1985) or RFC-1413 (St. Johns, 1993); and (2) strong authentication such as the Kerberos network authentication service (Neuman and Tso, 1994). RFC-931 and the more-recent RFC-1413 are connection-based applications on TCP that use an identification daemon running on each client host. This form of authentication has security limits. It is known, as noted in RFC-1413, that the information returned by these identification protocols is, at most, as trustworthy as the host providing it, and that if the host has been compromised, then the information obtained may be misleading or incorrect. Consequently, authentication using identification protocols, also called authentication by assertion, is used only to provide more auditing information for TCP connections. Stronger authentication methods based on cryptography are required. One such method is the Kerberos network authentication service, a secret-key authentication system developed at MIT under Project Athena. Version 5 (V5) of Kerberos is represented in RFC-1510 (Kohl and Neuman, 1993). Kerberos offers three different levels of protection: (1) authentication at the beginning of a network connection, (2) authentication for each message sent from one host to another, and (3) both authentication and encryption for each message sent from one host to another host. More-recent implementations of Kerberos support the Generic Security Service Application Program Interface (GSSAPI) described in RFC-1508.

Kerberos, however, has two fundamental limitations. First, Kerberos is not effective against password guessing attacks. That is, if a password is poorly chosen, an attacker may be able to guess and impersonate the user. Second, Kerberos requires a trusted path for password entry. If an attacker can monitor the path between the user and the initial program, then the user can be impersonated. Therefore, other tools, such as one-time passcodes and public-key cryptography, should be combined with Kerberos to improve security (Neuman and Tso, 1994). Although Kerberos is based on conventional

symmetric-key cryptography (Data Encryption Standard DES), recent extensions allow integration with public-key cryptography systems.

Most operating systems support distributed file systems such as the Network File System (NSF) and/or the Andrew File System (AFS), and some even support the Open Software Foundation (OSF) Distributed File System (DSF). These distributed file systems allow consistent data access on all machines across the network. Furthermore, DFS also provides security under the OSF Distributed Computing Environment (DCE) via credential transfer or other authentication services.

d. Provide Scheduling API for external scheduler

For any WMS package, no matter how many scheduling policies exist, or how flexible they are, eventually, an external scheduler must be inserted. The two most common reasons for having an external scheduler are:

- To enforce a specific scheduling policy that is not available in the WMS package. For example, Argonne National Laboratory and Cornell Theory Center inserted the Extensible Argonne Scheduling sYstem (EASY), into LoadLeveler to enforce a back-filling scheduling policy on the IBM SP-2 (Lifka, 1995; Skovira, 1996; Prenneis, 1996).
- To experiment with a research scheduler. For example, the Distributed Object Computation Testbed (DOCT) at San Diego Supercomputer Center planned to plug the research scheduler, Application Level Scheduler (AppLes), developed at the University of California, San Diego, into the WMS package, NQE (San Diego Supercomputer Center, 1997).

e. Support parallel and interactive jobs

Parallel applications are mostly programmed using the message-passing model. The two most widely used message-passing libraries are the Message Passing Interface (MPI) and the Parallel Virtual Machine (PVM). PVM had been the de facto leader before the message passing standardization effort. Since the ratification of MPI as the industry standard, many applications using the PVM library have been converted using MPI calls.

At a higher level, High Performance Fortran (HPF) is the standardized extension of Fortran 90 that supports parallel processing. HPF has been advancing in scientific and engineering applications. WMS should support HPF, if needed.

Support for interactive jobs are critical for software development, especially in the debugging stage. Many traditional database applications also require interactive-processing support. In addition, a growing number of organizations are making data available for Intranets, accessing information by way of web browsers or web-based applications. These interactive uses are expected to expand in the future.

f. Support a parallel job that requires a specified number of processors allocated to each host

This functionality allows the most flexible use of hosts that are multiprocessors. At one extreme, a parallel job may be scheduled onto a single multiprocessor host to take advantage of its efficient shared memory. At the other extreme, it may be spread out with one process on one host to take advantage of aggregate memory or to swap space or parallel I/O. However, it may be scheduled somewhere between the two cases when the former case is not possible. For example, in a cluster consisting of single-processor and dual-processor hosts, a parallel job running on four processors cannot be scheduled to a single host but may be run on two dual-processor hosts.

g. Provide statistical reports on system resource usage by users on different hosts

Statistical reports on usage of system resources by users on different hosts under various queues can be processed from job logs. Statistics should include the average, minimum, maximum, and total of common measures such as number of jobs, throughput, turn-around time, wait time, CPU time, user time, system time, memory, and swap space. These reports are valuable for system performance assessment, workload characterization, resource bottleneck identification, and system upgrade planning.

h. Provide system performance prediction and feedback to scheduler

Although the reports noted above are useful, they are static. Preferably, the WMS should also provide dynamic system-load data collected for short time intervals to support system performance prediction and feedback to the scheduler. Forecasts for network performance (latency and bandwidth) as well as system performance (CPU usage) for each host are of considerable interest to dynamic scheduling and developing quality-of-service (QoS) guarantees (Wolski, 1997).

i. Support checkpointing, restart, and job migration

Checkpointing is a useful means to save the state of a job (at regular time intervals) so it can be restarted later. Checkpointing is essential for implementing fault tolerance and load balancing. Indeed, if the execution host goes down, the checkpointed job can be restarted, on the same execution host when it becomes available or on a different host, at the last checkpoint instead of the beginning. However, a checkpointed job running on an overloaded host can be restarted on an idle or lightly loaded host. Job migration enhances load balancing.

There are two types of checkpointing: system-initiated and user-initiated. System-initiated (also called OS-level or kernel-level) checkpointing is automatically performed when requested by the submitted job without requiring the code to be modified and linked with any special library. User-initiated checkpointing can be either user-level checkpointing or application-level checkpointing. At the user level, application programs need to be linked with checkpoint libraries, whereas at the application level, programs are explicitly coded using checkpointing function calls and need to be linked with checkpoint libraries.

Although the usefulness of checkpointing is well-understood, the current state of implementation shows many limitations:

- Checkpointing of parallel jobs is complicated by network activity that causes difficulty in capturing the state of a parallel application. WMS packages mostly support checkpointing for serial jobs.
- Checkpointing inherently requires a lot of disk space because it creates a checkpoint file (snapshot of the application at the checkpoint time) that will be coupled with the original executable to form a new user executable that is needed when restarting the job. Checkpoint files are large, and writing checkpoint files to disk not only creates an overhead but may have impact on the file system performance and network bandwidth. Consequently, the cost of checkpointing may outweigh the benefit for (serial) jobs that run in a short time.
- The checkpointed job can only be restarted on a host that has the same architecture and runs the same operating system as the host on which the checkpoint was created.

- To prevent unpredictable results, jobs intended for checkpointing must be statically linked, cannot use internal timers, and should limit the use of other system calls for signals, forks, shared memory, semaphores, messages, and threads.

Consequently, the status of checkpoint implementation in WMS packages suggests that it is suited and useful for serial jobs that are long-running and compute-intensive.

j. Support Windows NT and allow a mixed NT/Unix environment

Market momentum for Windows NT continues to build, as shown by the acceptance of Windows NT as the chosen operating system for new networked PCs. The mixed NT/Unix environment will soon be common, considering the realization of programs such as the Navy's Information Technology for the 21st Century (IT21) initiative. To strategically position itself, a WMS package must exploit this heterogeneous environment.

3.4 CHALLENGES

There are many challenges for WMS to efficiently support a large-scale, heterogeneous, distributed computing environment. In particular, one can find several major challenges from the list of the 10 desirable features in the last section. Two issues of immediate importance are:

a. Checkpointing. The goal to provide universally available checkpoint services for serial and parallel jobs in a wide-area, heterogeneous, distributed computing environment remains elusive. Difficulties arise in the heterogeneity of the environment. Differences in vendor implementation of operating systems, different architectures, and various support levels for user-initiated checkpointing make providing ubiquitous checkpoint services a challenging task (Livny and Raman, 1998). While checkpointing for serial jobs has made great advances (Litzkow, Tannenbaum, Basney, and Livny, 1997), checkpointing for parallel jobs (especially for applications using the industry-standard MPI) are still in the research stage (Pruyne and Livny, 1996).

b. Interoperability. A large-scale distributed computing environment typically spans several sites under different administrative domains. Resources of these sites are often heterogeneous and under different WMS, provided mostlyby platform vendors. At present, WMS packages are not interoperable. To interface with various WMS packages is a great challenge to wide-area, heterogeneous, distributed computing.

While checkpointing is the most difficult goal to achieve for practical purposes (Livny and Raman, 1998), recently there has been design and experimentation of tools to provide workable interfaces to different WMS (Czajkowski, et al., 1997; Brunett and Fitzgerald, 1998). The next section discusses this effort and other recent developments related to WMS and wide-area, heterogeneous, distributed computing. These developments, although still in preliminary stages, are expected to have a profound impact in the future.

3.5 CONCLUSIONS

With the emergence of the concept of distributed metacomputing and the advent of high-performance, broadband networking, the focus is on WMS that is expected to transform a collection of geographically dispersed, administratively separated, heterogeneous platforms into virtually a single, shared-computing resource. WMS has made good progress in recent years. However, its current incapability is well-recognized. A short list of the 10 most desired features was presented in

section 3 as a general guide for choosing a WMS package. The following references provide the details for specific requirements.

3.6 BIBLIOGRAPHY

1. LoadLeveler, LSF, NQE, and PBS provide up-to-date, comprehensive documentation for online viewing and download.

   - LoadLeveler Documentation
     Current version 1.3, released in August 1996
     http://www.rs6000.ibm.com/resource/aix_resource/sp_books/loadleveler/index.html

   - LSF Documentation
     Current version 3.2, released in August 1998
     http://www.platform.com/content/products/documentation/default.html

   - NQE Documentation
     Current version 3.3, released in March 1998
     http://www.cray.com/products/software/nqe/documentation.html

   - PBS Documentation
     Current version 1.1.12, released in July 1998
     http://science.nas.nasa.gov/Software/PBS/docs.html

2. A seminal book titled "The Grid: Blueprint for a New Computing Infrastructure," edited by Ian Foster and Karl Kesselman, published by Morgan-Kaufmann Publishers, offers a good source on the state of the art in wide-area, high-performance distributed computing. The terminology "grid" refers to the analogy with electrical power grid to convey the notion that computational grids have the potential to provide access to computational resources in a pervasive, inexpensive, and dependable manner.

3.7 REFERENCES

Baker, M., G. Fox, and H. Yau. 1995. *Cluster Computing Review*. A Report for the British Joint Information Systems Committee (JISC) New Technology Sub Committee (NTSC). Also appeared as NPAC Technical Report SCCS-748, Northeast Parallel Architectures Center, Syracuse University, Syracuse, NY, Nov.

http://www.npac.syr.edu/techreports/html/0700/abs-0748.html

Available in HTML and PostScript formats.

Also appeared as an online article in the DARPA-funded National High-

Performance Software Exchange (NHSE) Review, 1996 Volume, First Issue,

htttp://nhse.cs.rice.edu/NHSEreview/CMS/

Brunett, Sharon, and Steven Fitzgerald. 1998. "Metacomputing Supports Large-Scale Distributed Simulations." To appear in *Proceedings of Supercomputing 98*. November, Orlando, FL.

http://www.cacr.caltech.edu/sfexpress/sc98.html

Czajkowski, et al. 1997. "A Resource Management Architecture for Metacomputing Systems." Paper presented at *IPPS/SPDP '97 Workshop on Job Scheduling Strategies forParallel Processing*. April, Geneva, Switzerland.

ftp://ftp.globus.org/pub/globus/papers/gram97.pdf

Jones, J. 1996a. *NAS Requirement Checklist for Job Queuing/Scheduling Software*. NAS Systems Division Technical Report NAS-96-003, NAS Systems Division, NASA Ames Research Center, Moffett Field, CA, April.

http://science.nas.nasa.gov/Pubs/TechReports/NASreports/NAS-96-003/

Available in HTML, FrameMaker, and PostScript formats.

Jones, J. 1996b. *Evaluation of Job Queuing/Scheduling Software: Phase 1 Report*. NAS Systems Division Technical Report NAS-96-009. NAS Systems Division, NASA Ames Research Center, Moffett Field, CA, July.

http://science.nas.nasa.gov/Pubs/TechReports/NASreports/NAS-96-009/

Available in FrameMaker, and PostScript formats.

Jones, J. and C. Brickell. 1997. *Second Evaluation of Job Queuing/Scheduling Software: Phase 1 Report*. NAS Systems Division Technical Report NAS-97-013. NAS Systems Division, NASA Ames Research Center, Moffett Field, CA, June.

http://science.nas.nasa.gov/Pubs/TechReports/NASreports/NAS-97-013/

Available in HTML, FrameMaker, and PostScript formats.

Kaplan , J. and M. Nelson. 1994. *A Comparison of Queuing, Cluster and Distributed Computing Systems*. Technical Report NASA TM 109025 (Revision 1). NASA Langley Research Center, Hampton, VA, June.

http://techreports.larc.nasa.gov/ltrs/94/tm109025.tex.refer.html

Available in PostScript and PDF formats.

Kohl, J. and B. Neuman. 1993. *RFC 1510: The Kerberos Network Authentication Service.* Internet Engineering Task Force. September.

ftp://ftp.isi.edu/in-notes/rfc1510.txt

Litzkow, M., T. Tannenbaum, J. Basney, and M. Livny. 1997. *Checkpoint and Migration of UNIX Processes in the Condor Distributed Processing System.* University of Wisconsin-Madison, Department of Computer Sciences Technical Report 1346, April.

http://www.cs.wisc.edu/condor/doc/ckpt97.ps

Lifka, D. 1995. "The ANL/IBM SP Scheduling System." Paper presented at *IPPS'95 Workshop on Job Scheduling Strategies for Parallel Processing*, April, Santa Barbara, CA.

http://www.tc.cornell.edu/Papers/lifka/ipps95.ps

Livny, Miron, and Rajesh Raman. 1998. "High-Throughput Resource Management." *The Grid: Blueprint for a New Computing Infrastructure.* Edited by Ian Foster and Carl Kesselman. Morgan-Kaufmann Publishers. San Francisco, CA.

Neuman, B. and T. Tso. 1994. "Kerberos: An Authentication Service for Computer Networks," *IEEE Communications Magazine*, Vol. 32, No. 9, September, pp. 33-38.

http://nii.isi.edu/publications/kerberos-neuman-tso.html

Papakhian, Mary. 1998. "Comparing Job-Management Systems: the User's Perspective." *IEEE Computational Science & Engineering*, Vol. 5, No. 2, April-June, pp. 4-9.

Prenneis, A. 1996. "LoadLeveler: Workload Management for Parallel and Distributed Computing Environments," paper presented at *SUPEUR,* Oct.

ftp://ftp.austin.ibm.com/pub/lscftp/papers/Supeur.paper.ps

Pruyne, J., and M. Livny. 1996. "Managing Checkpoints for Parallel Programs." Paper presented at *IPPS'96 Workshop on Job Scheduling Strategies for Parallel Processing*, April, Honolulu, HI.

http://www.cs.wisc.edu/condor/doc/ckpt_mgmt.ps

San Diego Supercomputer Center. 1997. *Evaluation of Queuing/Loadleveling Systems for DOCT.* A Distributed Object Computation Testbed (DOCT) White Paper. January, San Diego, CA.

http://www.sdsc.edu/DOCT/Publications/f1-1/f1-1.html

St. Johns, M. 1985. *RFC 931: Authentication Server.* Internet Engineering Task Force. January.

ftp://ftp.isi.edu/in-notes/rfc931.txt

St. Johns, M. 1993. *RFC 1413: Identification Protocol.* Internet Engineering Task Force. February.

ftp://ftp.isi.edu/in-notes/rfc1413.txt

Skovira, J., W. Chan, H. Zhou, and D. Lifka. 1996. "The EASY-LoadLeveler API Project." Paper presented at *IPPS'96 Workshop on Job Scheduling Strategies for Parallel Processing*, April, Honolulu, HI.

http://www.tc.cornell.edu/Papers/lifka/ipps96.ps

Wolski, R.. 1997. "Forecasting Network Performance to Support Dynamic Scheduling Using Network Weather Service." *Proceedings of the 6th IEEE Symposium on High Performance Distributed Computing (HPDC),* August, Portland, OR, pp. 316-325.

ftp://ftp.cs.ucsd.edu/pub/rich/papers/nws-hpdc.ps.gz

4. AGENTS, WORKLOAD MANAGEMENT, AND WIDE-AREA, HETEROGENEOUS, DISTRIBUTED COMPUTING

4.1 INTRODUCTION

Intelligent agents have moved from the abstract notions of artificial intelligence to many diverse implementations fostered by the explosive growth of the Internet and electronic commerce. These software agents (or simply, agents) are touted as able to book airline tickets, monitor stock quotes, and perform many other daily transactions as well as to mine data for competitive enterprises and organizations. Agents, now recognized as a booming field of distributed computing, provide a new framework to facilitate many services and applications.

In general, there are two types of agents: stationary (or anchored) agents and mobile (or transportable) agents. While a stationary agent can run on only one machine (either on a client or on a server), a mobile agent can roam from one machine to another. Agents provide a framework facilitating wide-area, heterogeneous, distributed computing, especially easing the workload-management task. In addition, mobile agents could provide an efficient environment for mobile computing.

In this section, we present an overview of agents' roles in wide-area, heterogeneous, distributed computing, and discuss advantages and challenges of the emerging technology. The section is structured as follows: in section 4.2, we motivate the presentation with two mobile-agent distributed computing models and give a comparison. Section 4.3 discusses recent trends and developments in workload-management software. Of particular interest are (1) the explicit adoption of agents in scheduling at least at the research level, (2) the use of a system performance monitoring service to facilitate dynamic scheduling and fault detecting, and (3) the effort to coordinate resources under different local workload-management software. Conceptually, agent-based monitoring and management can help mitigate architectural complexity and enhance performance. Section 4.4 discusses the advantages and challenges of agent-based approaches. Finally, section 4.5 presents a summary and conclusion.

4.2 TWO MOBILE-AGENT DISTRIBUTED COMPUTING MODELS

To facilitate discussion, we will describe two models for distributed computing using mobile agents. Both models are based on the client-server paradigm and the classic Master-Slave pattern. The main difference between the two models is the degree of the agents' autonomy with respect to mobility. The overall implication is nontrivial as noted in comparison.

4.2.1 Two Models

The first model was described in a tutorial (Sommers, 1997) that appeared in the online magazine, *JavaWorld*, in April 1997, and that helped popularize the use of mobile agents in distributed computing. The article described how to create mobile agents in Java using IBM's Aglets Workbench (now called Aglets Software Development Kit) to build two applications: a distributed search engine and, more interestingly, a virtual supercomputer. The latter is described in this section.

As noted in Sommers (1997), the idea of harvesting the processing power of a virtual supercomputer composed of Internet-connected machines via mobile agents had been exposed (incidentally) in another author's article (Vanhelsuwe, 1997), also in *JavaWorld*, 2 months earlier. Essentially, a virtual supercomputer can be built using the standard Master-Slave model where the Master spawns one or more slaves to perform a specific task. To facilitate load balancing, a Slave called LoadGatherer is tasked to provide the Master with load information so that the Master can redistribute spawned Slaves whenever load conditions change dramatically. Therefore, in this model, besides specifying the task performed by each Slave, the Master also performs the function of workload-management software.

Although in the article (Sommers, 1997) these agents (Master, Slaves, and LoadGatherer) were implemented in Java (called aglets, abbreviation of agent applets), the overall architecture is language-independent. Java is mostly preferred in implementing agent applications simply to ensure platform independence.

The agents' mobility facilitates redistributing of Slaves. Furthermore, mobility allows a more possibilities in implementing the LoadGatherer, thus promoting a better load distribution.

In addition, of particular interest is the possibility that mobile agent technology will enable the class of mobile clients to participate in wide-area, heterogeneous, distributed computing. Devices of this class, ranging from laptop and notebook (as well as the forthcoming palm-sized) computers to handheld personal digital assistants (PDAs), can get relevant results by launching mobile agents to perform necessary computation and filtering on high-performance platforms (Chess et al., 1995; Chess et al., 1994).

This mobility of agents, however, can also create security threats since a client can implant malicious tasks into agents or change their states. Security concerns have been recognized in deploying mobile-agent applications (for example, see Chess et al. [1994]) and Java-based mobile-agent applications (for example, see Karjoth, Lange, and Oshima [1997]). Many security issues have not been resolved and security mechanisms are among currently active research topics. For example, while authenticators (algorithms to determine an agent's authenticity) for the one-hop mode of mobility are well studied and understood, the multi-hop authentication has not been solved. This complicates the current security situation because the agent transfer (also called agent transport) is considered a muti-hop operation (Crystaliz, General Magic, GMD FOKUS, IBM, and the Open Group, 1997). In a closed distributed computing environment, the principal concern is not much on malicious attacks but mainly on accidents that resulted from errors when deploying mobile-agent-enabled software. Note that effects of accidental security violations, not different from intentional ones, could be detrimental to the safety and integrity of system resources including databases.

Since the LoadGatherer checks workload conditions and reports them to the Master, it does not contribute to the overall-processing task performed by other Slaves. Another alternative to this model is to abolish the LoadGatherer. Functionally, all agents will set out to find resources at runtime to perform their specified tasks. In this case, the Master, no longer engaging in the workload-management task, participates in the overall processing and, if necessary, coordinates with Slaves for gathering result and I/O task.

In a mobile-agent-based system, a mechanism for dynamic measurement of resource and network utilization, and for dissemination of information to agents, is needed. Above all, cooperation between agents is essential to ensure that agents are migrating efficently from heavily loaded to lightly loaded nodes. Without cooperation, mass migration of agents ever-seeking rescheduling could result in chaotic and dangerous behavior ("thrashing") as well as waste of resources.

One experimental approach based on microeconomic models has been explored in Bredin, Kotz, and Rus (1998a) and Bredin, Kotz, and Rus (1998b). In this approach, each resource has a resource manager that sets the price for the resource under its control. Pricing is fixed at initialization and is dynamically changing to adapt to supply-demand conditions of resources in the systems. One relevant model is seller-driven, characterized by many buyers competing for the same sources. Each agent has a pre-allocated budget for bidding. In addition, there is a model for option trading to counter market fluctuations.

A more traditional approach to an adaptive placement framework was recently described in Keren and Barak (1998). The load index of a node, based on the node's utilization, is communicated among agents so that at each time unit, each agent has information about a subset of nodes in the system. The migration decision is executed periodically by each agent. When the expected performance gain is above a predetermined threshold, migration follows after a reservation message is communicated to prevent concurrent migration.

A more sophisticated facility to provide dynamic forecasts of network and resource load conditions can be incorporated. The agent-based system, AppLeS, the application-level scheduler, uses a separate facility called the Network Weather Service (NWS) that parameterizes performance models to predict the state of resources at the time the application will be scheduled. Besides the usual resource-availability parameters, NWS also includes network-performance parameters such as bandwidth and latency. Both AppLeS and NWS have been developed at the University of California, San Diego (Berman, 1998; Berman and Wolski, 1997; Berman et al., 1996; Spring and Wolski, 1998).

4.2.2 Comparison

For convenience, we denote the first mobile-agent distributed computing model (Master-Slave-LoadGatherer), Model A, and the second model (Master-Slave), Model B. The implications of the degree of autonomy variation granted to the agents in Models A and B can be summarized as follows:

- Model B has the advantage of avoiding resource allocation for the tasks of gathering and reporting load conditions and workload management required in Model A. However, these services must be assumed by individual agents or provided as part of the infrastructure. Without these services, the possible performance gain might not be realized but performance degradation might happen to the overall task.
- Services leading to self-scheduling in Model B could be streamlined using fewer features supported by a reduced set of resource and network parameters so that each agent can adapt with ease. This is workable for a small and homogeneous distributed computing environment. For a large-scale, heterogeneous environment, simplifying resource and network parameters will ignore many complexities and result in performance problems. Much of the difficulty is caused by the heterogeneity of resources and the impact of variations in deliverable resource performance because of the contention of shared resources (Berman, 1998).

4.3 RECENT TRENDS AND DEVELOPMENTS IN WORKLOAD-MANAGEMENT SOFTWARE

Perhaps the most noticeable trend in WMS is the emergence of the use of agents at least at the research level. Clearly, widely used commercial WMS packages such as LoadLeveler, LSF, and NQE, that were designed and implemented before the popularity of agent technologies, have many components that can now be articulated as, in fact, intelligent agents. In this section, we present two re-

search projects explicitly designed with the use of agents for resource allocation and scheduling, one at the system level and the other at the application level. Another trend is the use of a system monitoring service aimed at improving dynamic scheduling and enhancing fault-detecting capability. In addition, there is an effort to coordinate resources under different WMS (such as LoadLeveler, LSF, and NQE). Since these local WMS packages are not interoperable, this endeavor is of utmost importance for creating a wide-area metacomputing environment.

4.3.1 Agent-based Scheduling

Agent-based scheduling can provide robust and high-performance application execution in a highly distributed computing environment. Both agent-based systems described in this section are stationary and the agents are performing dynamic scheduling.

One of the examples of agent-based scheduling is the NetSolve System developed jointly at the University of Tennessee and the Oak Ridge National Laboratory (Casanova and Dongara, 1998; Casanova, Dongarra, and Seymour, 1998; Casanova and Dongarra, 1997a; Casanova and Dongarra, 1997b). NetSolve is a client-server application that enables users to solve complex scientific problems using computational resources on the Internet, or on an Intranet, or by belonging to a research group. Numerical software currently available in the NetSolve system includes ARPACK, FFTPACK, LAPACK, BLAS, Minpack, QMR, and the parallel library, ScaLAPACK. Many interfaces to Fortran, C, Matlab, and Java have been designed and implemented to help users access hardware and software resources in the NetSolve system from their program. There is also a Java graphical user interface (GUI) based on Java Development Kit (JDK) 1.1 for user-friendly interfacing.

The NetSolve system uses a remote computing method where data from the client (user's machine) is sent to the server for execution, and the result is sent back to the client. The NetSolve agent performs the task of a resource broker for the clients. It takes the request from a client and makes the choice of resources in the NetSolve systems for execution of the client's program based on computation-specific and resource-specific information (Casanova and Dongarra, 1997a).

In addition, the NetSolve agent performs load balancing and fault tolerance (Casanova and Dongarra, 1997a). For load balancing, the NetSolve agent estimates the execution time of the client's problem on every machine in the NetSolve system and chooses the one(s) of smallest time. The execution time estimate is based on a performance model that considers (1) client-dependent parameters (such as the size of data to send, the size of result to be received, and the size of the problem); (2) static server-dependent parameters (such as the raw performance of servers, the complexity of algorithm, and network characteristics between the client and servers); and (3) dynamic server-dependent parameters (such as workload). Note that network characteristics (such as bandwidth and latency) between a client and a server is assumed stable (Casanova and Dongarra, 1997b).

The NetSolve agent uses two mechanisms to promote fault tolerance. First, NetSolve maintains a client-server protocol so that whenever the NetSolve agent receives a request for a problem to be solved, it sends back a list of computational servers sorted from the most to the least suitable one. The client tries all the servers in sequence until one accepts the problem (that is, establishes the connection). If no connection is received, the client will receive a different list or no server remains. Secondly, once the connection is established, if the server dies for some reason, the problem is migrated to another possible server until it is solved or no server remains (Casanova and Dongarra, 1997b).

Note that there are multiple instances of the NetSolve agent on the network, especially when the set of computational resources spans several local-area networks (Casanova, Dongarra, and Seymour, 1998). In this sense, NetSolve can be characterized as a multi-agent system.

Another agent-based scheduler is the Application-Level Scheduler (AppLeS), developed at the University of California, San Diego. While NetSolve schedules requests (and provides access to numerical software), AppLeS schedules stand-alone parallel distributed applications. For each application, AppLeS has a single active agent coordinating four subsystems. One of the subsystems is the NWS that provides performance monitoring and predictive functionality to the AppLeS agent. The other subsystems are Selector (selecting resources), Planner (generating a resource-dependent schedule), and Actuator (implementing the "best" schedule). The AppLeS agent uses (1) user preferences (for machines, libraries, performance measure), and (2) application-specific and dynamic information to determine a performance-efficient schedule. In the default scheduling policy, the schedule given by AppLeS is the "best" of the schedules determined; however, the user can override it by using his/her own scheduling policy. Details of the AppLeS design can be found in Berman (1998), Berman and Wolski (1997), and Berman et al. (1996).

Although the agent-based AppLeS scheduler is still in the development stage, it has generated interesting lessons learned from each new application (Spring and Wolski, 1998). There is a proposed collaborating effort between NetSolve and AppLeS to further improve the performance of the NetSolve system (Berman, 1998).

Overall, agent-based programming has implication on WMS. WMS can be agent-based, that is, WMS functionality can be performed by system-level agents. This is the case of the NetSolve scheduling agent(s) that can provide access to network resources for numerical-computing execution in a grid environment. However, the scheduling functionality is assumed by application-level agent(s) for the user's application. In the Application-Level Scheduler (AppLeS), the scheduling agent is a stationary agent that coordinates four subsystems to achieve the best scheduling. As described in section 4.2.1, other examples are mobile-agent models that either let mobile agents autonomously seek resources and migrate by themselves, or use model-based mechanisms (for example, a microeconomic, market-oriented approach) for resource control. In general, to achieve efficient end-to-end resource utilization, it requires coordination between system-level scheduling and application-level scheduling where an application-level scheduler can specify the application's performance requirements.

4.3.2 System Performance Monitoring

Two immediate benefits of monitoring system performance are to dynamically forecast network and computational resource load and availability, and to detect faults. The two monitoring services currently available are the Network Weather Service (Wolski, 1997) that is used in AppLeS (Berman, 1998), and the HeartBeat Monitor (Sterling et al., 1998), a service in the Globus system, that has been integrated into NetSolve as an option for an application-specific restart-based fault recovery mechanism (Casanova, Dongarra, and Seymour, 1998).

The Network Weather Service (NWS) facility provides AppLeS with dynamic forecast of resource load and availability. As noted above, bandwidth and latency are assumed stable and treated as static parameters in the NetSolve system. The NWS, however, considers them dynamic parameters, and periodically monitors and dynamically forecasts the performance of network and computational resources in terms of these and other parameters. The current methods used in forecasting are mean-

based, median-based, and autoregressive. The system tracks the accuracy of these three predictors and uses the one of lowest cumulative error at any given time to generate a forecast (Wolski, 1997).

While the network sensory subsystem implementation of the NWS is a modification of the well-known network performance probing utility, *netperf*, the Globus HeartBeat Monitor (HBM) was a new design. Essentially, the HBM has two components. One is the client registration API that allows a process to register with the HBM, and then expects to receive regular heartbeat from the process. The other component is the data collection API that allows another process to get information about the status of a registered process. Developed as a part of core Globus services, HBM checks the status of other core services such as resource allocation manager, directory service, and implements application-specific, fault-recovery strategies (Sterling et al., 1998).

Conceptually, system-performance monitoring can be implemented as an agent-based service. In particular, the monitoring service can be decomposed into tasks performed by autonomous, adaptable, agent-based monitors that communicate constantly with peers and resources through an agent-oriented communication mechanism (Johnston, 1998).

4.3.3 Interoperability

Typically a large-scale distributed computing environment is heterogeneous with resources geographically dispersed, and local sites are often under different workload-management software packages. Frequently, WMS packages are bundled with platforms from hardware vendors. For example, Cray and SGI platforms use NQE; IBM platforms (RS-6000 and SP-2) use LoadLeveler or LoadLeveler with the EASY scheduler; and Sun and others use LSF (which has been gaining market share because of its wide coverage of supported platforms). Since these local resource-allocation mechanisms do not interoperate, there is a serious problem in resource management and scheduling for wide-area distributed computing.

The ideal way to solve this problem is to standardize resource management and scheduling APIs as explored in the PSCHED initiative (PSCHED, 1997). This effort has been inactive for more than a year, and there is no sign that the proposed APIs will ever be completed because of the main vendors' lack of interest. A less than ideal but possible solution is to devise higher level resource allocation managers that interface with these (and other) local resource allocators. The Globus Resource Allocation Managers (GRAMs), a core service of the Globus toolkit, uses this method to map the resource specification into a request to some local resource allocators. Currently, GRAM can interface with six different resource allocation mechanisms: LoadLeveler, LoadLeveler with EASY scheduler, LSF, NQE, Condor, and a simple "fork" daemon (Czajkowski et al., 1997).

An extensible resource specification language (RSL), based on the syntax for filter specification in the Lightweight Directory Access Protocol (LDAP), communicates the request for resources between components of the Globus resource management architecture. To facilitate access to information on the current availability and capability of resources, the Metacomputing Directory Service provides a suite of tools and APIs based on LDAP to facilitate access to information on the current availability and capability of resources (Czajkowski et al., 1997).

Conceptually, the task to interface local WMS can be performed by agent(s) to mitigate architectural complexity. In particular, GRAMs as well as other subsystems of the Globus resource-management architecture can be agent-based. Again, an effective mechanism for agent communications is required.

The overall Globus resource-management architecture were implemented and deployed on GUSTO, a large computational grid testbed comprised of 15 sites and over 330 computers using LSF, NQE, LoadLeveler, LoadLeveler with EASY, and Condor as local WMS (Czajkowski et al., 1997). The DARPA-sponsored Synthetic Forces (SF) Express project is one defense-related research application that has experimented with the Globus toolkit. In Mach 1998, at the Technology Area Review Assessment (TARA) briefing in San Diego, the SF Express project showed a distributed, parallel implementation of the widely used Modular Semi-Automated Forces (ModSAF) Distributed Interactive Simulation with 100K entities, spanning nine sites and over 1300 processors. This demonstration set a new record with respect to the number of entities in Distributed Interactive Simulation (Brunett and Fitzgerald, 1998)

4.4 ADVANTAGES/CHALLENGES OF AGENT APPROACH

Perhaps the most far-reaching advantage of agents is in the intelligent integration of information. Individuals in a modern society are facing ever-expanding resources that lead to information overload. The same situation occurs in large-scale distributed computing. The trend is that the computing environments are getting wider, encompassing multiple administrative domains having many heterogeneous resources (including information resources). Practical advantages of agents are at both application and system levels. At the application level, agents help get optimal performance by using the best-suited resources available in the system. At the system level, agents promote load balancing.

Addtionally, it is believed that mobile agents reduce network load. In traditional client-server distributed systems, communications protocols often need multiple interactions to complete a particular task, especially when large volumes of data are located remotely. Mobile agents allow the processing to be completed where data are located by migrating code to data. Another consequence of code mobility is that mobile agents overcome network latency.

Mobile agents also have important implications on the class of mobile platforms. On one hand, these platforms usually have limited processing capacities due to relatively slow processors and limited memory as well as limited-capability software and environments. On the other hand, they mostly operate under unforgiving network conditions. In particular, their connections often have low bandwidth and high latency, and are prone to failure because of inadvertent interference. Moreover, they do not have permanent connections into a network, and mostly, they are disconnected for a long period. Mobile agents are attractive for mobile platforms because they migrate and access resources in the network, and thus, do not need constant connection to the home platforms. Upon completing their tasks, mobile agents send results back to the mobile platforms. If a mobile platform is disconnected, results are temporary stored on a docking mechanism that re-sends when connection is re-established (Chess et al., 1995; Gray, Kotz, Nog, Rus, and Cybenko, 1996).

As in the case of many emerging technologies, there is a large gap between expectations for agents (in general) and mobile agents (in particular) and the available technology. Many technical challenges to agents (in general) and mobile agents (in particular) remain. First, agents need policy and mechanisms for cooperation. Without cooperation, an agent-based system is very likely subject to chaotic and dangerous behavior as well as waste of resources. The "thrashing" situation could happen in an agent-based system, and would be more likely in a mobile-agent-based system (Berman, 1998).

Second, agent-based systems need security mechanisms, especially for mobile agents that are able to roam from one machine to another. Security measures are needed to protect machines from agents

and agents from machines—against malicious and inadvertent attacks. Two techniques under investigation are code appraisal and state appraisal. Code appraisal aims to provide programming language support to improve mobile code safety. Developing a safe language for mobile agents or mobile code can accomplish this goal. Another possibility is using proof-carrying code that packages a proof together with the agent, and for the host to check the proof's validity. Because a migrating agent can become malicious if its state is altered or corrupted, state appraisal is also crucial to system security (Swarup, 1997; Farmer, Guttman, and Swarup, 1996).

Third, there is a great need for agent interoperability. In particular, agent-based systems need standardized methods for common operations including agent creation, deletion, and, for mobile agents, transfer across the network. Currently, the Mobile Agent Facility (MAF) Specification, drafted by IBM, General Magic, Crystaliz, and GMD FOKUS, with Open Group support, has been jointly submitted to the Object Management Group (OMG) for consideration as an industry standard (Crystaliz, General Magic, GMD FOKUS, IBM, and the Open Group, 1997). Standardization efforts are needed in other areas such as communication protocols and system integration.

Finally, the agent technology will benefit from efforts to provide cost-effective agent-development infrastructure including languages, tools, and environments. Wide-area, heterogeneous, distributed computing will, in turn, benefit from the agent technology. In particular, tools are needed to create resource- and network-aware agent-based applications. These applications must be adapted to dynamic system state and be able to assess the performance impact and select potential platforms. Development environments supporting debugging and environment tuning for agent-based applications are crucial because of the resource heterogeneity. In a wide-area, heterogeneous, distributed computing environment, the other users' impact on shared resources and system asynchrony make diagnostics and tuning extraordinarily difficult (Berman, 1998).

4.5 CONCLUSION

In this chapter, we present several fundamental developments in workload management in wide-area distributed computing. As discussed in section 3, WMS interoperability is crucial when resources are under different WMS. To mitigate the lack of WMS interoperability packages, one approach is to devise high-level interfaces to coordinate with local WMS packages. This approach was adopted in the design of a resource-management architecture of the Globus toolkit for wide-area, heterogeneous, distributed computing. Another development is the use of system-performance monitoring services to facilitate dynamic scheduling and fault detection.

It is believed that the rapidly growing agent technology is well suited to wide-area, heterogeneous, distributed computing because it can help enhance design and performance. In particular, agent-programming models can offer scheduling at the system level as well as the application level. Overall, to achieve efficient end-to-end resource utilization, it needs coordination between system-level scheduling and application-level scheduling where an application-level scheduler can specify the application's specific performance requirements. Furthermore, mobile agents provide better support for mobile computing. Mobile platforms—such as laptop, notebook, and palm-sized computers as well as handheld personal digital assistants (PDAs)—are characterized by limited storage and processing capacity, limited-capacity software and runtime environment, low-bandwidth high-latency connection, and long-period disconnection. Mobile agents enable this class of clients to get relevant results where the necessary computation and filtering are performed on high-performance servers

Many technical challenges to the agent technology remain. Several main issues have been presented in this section:

- Many security issues on deploying mobile-agent applications as well as Java-based mobile-agent applications have not been resolved (Chess et al., 1994; Karjoth, Lange, and Oshima, 1997). In particular, while the agent transfer is considered a multi-hop operation, the current security technology has not solved the multi-hop-authentication problem (Crystaliz, General Magic, GMD FOKUS, IBM, and the Open Group, 1997). In a closed distributed computing environment, the principal concern is not much on malicious attacks but mainly on accidents that resulted from errors (for example, software bugs) when deploying mobile-agent-enabled software. Note that the effects of accidental security violations, not different from intentional ones, could be detrimental to the safety and integrity of system resources including databases. Indeed, it was recommended that "*substantial investment in mobile agent systems may await further work on new security techniques oriented toward mobile agents*" (Swarup, 1996).

- Agent technology urgently needs a standardized specification. The wide differences in architecture and implementation among agent and mobile-agent systems currently available prevent interoperability, and thus, serious adoption. The Mobile Agent Facility (MAF) Specification, drafted by IBM, General Magic, Crystaliz, and GMD FOKUS, with Open Group support, was jointly submitted to the Object Management Group (OMG) for consideration. This effort covers common agent operations including agent creation, termination, and transfer. Mobile agents are, as noted in the current MAF draft, "*a relatively new technology that is fueling a new industry*" (Crystaliz, General Magic, GMD FOKUS, IBM, and the Open Group, 1997). Consequently, the sooner the specification is finalized, the better the chance of a sustained growth.

- Besides the security limitations of mobile agents, knowledge representation that facilitates agents in communications with other agents is another important issue that needs resolution to ensure a viable (Kiniry and Zimmerman, 1997). Current technologies (such as those from the DARPA-sponsored Knowledge Sharing Effort) are highly complex for integration into agent systems. Furthermore, present agent communication mechanisms are extremely limited in flexibility and possibilities (Kiniry and Zimmerman, 1997). Knowledge representation and communication enable applications in collaboration, intelligent planning and scheduling, electronic commerce, and many other areas.

- Agent technology is still in the early developmental stage. For workload management of a large-scale, heterogeneous, distributed computing environment, it can build a tools leveraging, well-developed infrastructure, as in the case of research-oriented projects such as the agent-based scheduler, AppLeS (together with the resource and network monitor and predictor, NWS). In general applications as well as in workload management applications, there has been a pervasive emphasis on the agent "smartness." However, cooperation and adaptive learning are critical to building powerful applications. Indeed, "individual agents needn't be so smart to function collectively in a complex and useful manner—just like an ant colony—and that agents can learn from one another" (Petrie, 1997).

- Programming engineering and scientific applications using agents (and mobile agents) have drawbacks similar to using the message-passing paradigm, except that the Message-Passing Interface (MPI) standardized the latter. In particular, agent programs are generally difficult to write and exceptionally difficult to debug. For example, in the Master-Slave pattern, it requires that the program's data structures and the entire application to be explicitly partitioned so each

Slave (here, a mobile agent) can perform its own specified subtask. Further complications are in the communications among Slaves that require synchronization. Except for coarse-grained tasks that are inherently data-independent or those that can decompose easily and do not need much communication, programming with agents is a complex endeavor, not unlike low-level language programming. Currently, agent technology is not mature enough for processing real-life, mission-critical tasks in a wide-area, heterogeneous, distributed environment. In the near-term, agent technology will benefit from a cost-effective agent-development infrastructure including languages, tools, and environments.

- The DARPA Information Systems Office (ISO) is sponsoring projects under control of the Agent-Based Systems (CoAABS) Program to develop and demonstrate techniques to safely control, coordinate, and manage large systems of agents. The ISO is also funding the Command Post of the Future (CPOF) Program where agent technology is one of the many candidates under assessment. More information can be found at the respective web pages listed in the General References section.

4.6 BIBLIOGRAPHY

IEEE Internet Computing frequently published articles on agent-based applications and infrastructure. In the special issue on "Internet-based Agents" (Vol. 1, No. 4, July-August 1997) there are many pointers to resources for agents and Java-based agents. The magazine has a regular column titled "Agents on the Web".

IEEE Internet Computing:
http://www.computer.org/internet/

IBM's Aglets Workbench, now called Aglets Software Development Kit (ASDK), is available as free software from IBM Tokyo Research Laboratory. It is a first-of-its-kind visual environment for creating mobile-agent-based applications in Java. Software and documentation are available at

IBM Aglets Software Development Kit:

http://www.trl.ibm.co.jp/aglets/

The DARPA Information Systems Office (ISO) has several agent-related programs, including

Control of Agent-Based Systems:

http://dtsn.darpa.mil/iso/programtemp.asp?mode=126

Command Post of the Future:

http://www-code44.spawar.navy.mil/cpof/

DARPA sponsored the three-day Workshop on Foundations for Secure Mobile Code in March 1997 at Naval Post-graduate School. Position papers presented at the workshop are available at

Workshop on Foundations for Secure Mobile Code:

http://www.cs.nps.navy.mil/languages/statements/

The ACM 1998 Workshop on Java for High-Performance Network Computing was held on the campus of Stanford University, Palo Alto, CA, February 28-March 1, 1998. This is the third of a series of workshop started in 1996 to explore the use of the Java programming language for high-performance network computing, and scientific and engineering computing. Proceedings of the 1998 workshop are available online at

ACM 1998 Workshop on Java for High-Performance Network Computing:

http://www.cs.ucsb.edu/conferences/java98/

Globus is an integrated software system for distributed parallel computing intended to create wide-area virtual computers. Recognizing that virtual computers must support a variety of applications and programming models, the Globus system adopts a mix-and-match approach by providing the Globus metacomputing toolkit from which users and developers can select to meet their need. The toolkit, embodied core technologies for computational grid, covers areas of resource management, security, communication, and data access.

Globus:

http://www.globus.org

An interesting defense application using the Globus toolkit is the DARPA-funded Synthetic Forces (SF) Express project at Caltech that has recently set a record in Distributed Interactive Simulation (DIS).

SF-Express:

http://www.cacr.caltech.edu/sfexpress/

The well-known *netlib* depository of freely available numerical software now offers the NetSolve system to provide user access to preinstalled numerical libraries that allow execution using remote computing resources of a wide-area virtual computer.

NetSolve:

http://www.netlib.org/netsolve/

The agent-based Application Level Scheduler (AppLeS) and the network and resource monitor Network Weather Service (NWS) have a homepage at

AppLeS:

http://www-cs.ucsd.edu/groups/hpcl/apples/apples.html

4.7 REFERENCES

Berman, Francine. 1998. "High-Performance Schedulers." *The Grid: Blueprint for a New Computing Infrastructure.* Edited by Ian Foster and Carl Kesselman. Morgan-Kaufmann Publishers. San Francisco, CA.

An early version of the material is available:

http://www.cs-ucsd.edu/users/berman/cse260/chapter.ps

Berman, Francine, and Rich Wolski. 1997. "The AppLeS Project: A Status Report." *Proceedings of the 8th NEC Research Symposium.* Berlin, Germany. May.

ftp://ftp.cs.ucsd.edu/pub/berman/nec.ps

Berman, Francine, Rich Wolski, Silvia Figueira, Jenifer Schopf, and Gary Shao. 1996. "Application-Level Scheduling on Distributed Heterogeneous Networks." *Proceedings of Supercomputing 96.* November.

http://www.cs.ucsd.edu/groups/hpcl/apples/sup96/apples.ps

Bredin, Jonathan, David Kotz, and Daniela Rus. 1998a. "Market-Based Resource Control for Mobile Agents." *Proceedings of the Second International Conference on Autonomous Agents.* May.

ftp://ftp.cs.darmouth.edu/pub/kotz/papers/bredin:market.ps.Z

Bredin, Jonathan, David Kotz, and Daniela Rus. 1998b. *Utility Driven Mobile-Agent Scheduling.* Technical Report PCS-TR98-331. Dartmouth College. May ftp://ftp.cs.dartmouth.edu/TR/TR98-331.ps.Z

Brunett, Sharon, and Steven Fitzgerald. 1998. "Metacomputing Supports Large-Scale Distributed Simulations." To appear in *Proceedings of Supercomputing 98.* November, Orlando, FL.

http://www.cacr.caltech.edu/sfexpress/sc98.html

Casanova, Henri, and Jack Dongarra. 1998. "Applying NetSolve's Network-Enabled Server." *IEEE Computational Science & Engineering.* Vol. 5, No. 3, July-September, pp. 57-67.

Casanova, Henri, and Jack Dongarra. 1997. "Using Agent-based Software for Scientific Computing in the NetSolve System." Submitted to *Parallel Computing.*

http://www.netlib.org/utk/people/JackDongarra/pdf/netsolve-agent.pdf

Casanova, Henri, and Jack Dongarra. 1997. "NetSolve: A Network-Enabled Server for Solving Computational Science Problems." *International Journal of Supercomputer Applications and High Performance Computing.* Vol. 11, No. 3, Fall 1997, pp. 212-223.

http://www.netlib.org/utk/people/JackDongarra/pdf/netsolve.pdf

Casanova, Henri, Jack Dongarra, and Keith Seymour. 1998. *Users' Guide to NetSolve version 1.1.b (Client and Server).* Univ. of Tennessee. May 14.

http://www.netlib.org/netsolve/ug.ps

Chess, David, Benjamin Grosof, Colin Harrison, David Levine, Colin Parris, and Gene Tsudik. 1995. "Itinerant Agents for Mobile Computing." *IEEE Personal Communications.* Vol. 2, No. 5, October, pp. 34-49.

Chess, David, Colin Harrison, and Aaron Kershenbaum. 1994. *Mobile Agents: Are They a Good Idea?* IBM Research Report RC-19887. Declassified March 16, 1995.

http://www.research.ibm.com/iagents/paps/mobile_idea.ps

Also appeared in *Mobile Object Systems: Toward the Programmable Internet.*

Lecture Notes in Computer Science No. 1222, pp. 25-45, Springer-Verlag, 1997.

Crystaliz, Inc., General Magic, Inc., GMD FOKUS, IBM, and the Open Group. 1997. *Mobile Agent Facility Specification. Joint Submission* (Draft 7). July 2.

http://www.infosys.tuwien.ac.at/Research/Agents/archive/special/mafdraft7.ps.gz

http://shinjuku.genmagic.com/~cynthia/mafeval/submission/draft7/draft7.ps

Czajkowski, et al. 1997. "A Resource Management Architecture for Metacomputing Systems." Paper presented at *IPPS/SPDP '97 Workshop on Job Scheduling Strategies for Parallel Processing*. April, Geneva, Switzerland.

ftp://ftp.globus.org/pub/globus/papers/gram97.pdf

Farmer, William, Joshua Guttman, and Vipin Swarup. 1996. "Security for Mobile Agents: Issues and Requirements." *Proceedings of the 19th National Information Systems Security Conference*, Baltimore, MD, October.

http://csrc.nist.gov/nissc/1996/papers/NISSC96/paper033/SWARUP96.pdf

Gray, Robert, David Kotz, Saurab Nog, Daniela Rus, and George Cybenko. 1996. *Mobile Agents for Mobile Computing*. Technical Report PCS-TR96-285. Dartmouth College, May.

ftp://ftp.cs.dartmouth.edu/TR/TR96-285.ps.Z

Johnston, William. 1998. "Realtime Widely Distributed Instrumentation Systems." *The Grid: Blueprint for a New Computing Infrastructure.* Edited by Ian Foster and Carl Kesselman. Morgan-Kaufmann Publishers. San Francisco, CA.

Karjoth, Gunter, Danny Lange, and Mitsuru Oshima. 1997. "A Security Model for Aglets." *IEEE Internet Computing*. Vol. 1, No. 4, July-August, pp. 68-77.

Keren, Arie, and Amnon Barak. 1998. "Adaptive Placement of Parallel Java Agents in a Scalable Computing Cluster." Paper presented at *ACM 1998 Workshop on Java for High-Performance Network Computing*. February 28-March 1. Palo Alto, CA.

http://www.cs.ucsb.edu/conferences/java98/papers/agents.ps

Kiniry, Joseph, and Daniel Zimmerman. 1997. "A Hands-On Look At Java Mobile Agents." *IEEE Internet Computing*. Vol. 1, No. 4, July-August, pp. 21- 30.

Petrie, Charles. 1997. "What's an Agent ... And What's So Intelligent About It?" *IEEE Internet Computing*. Vol. 1, No. 4, July-August, pp. 4-5.

PSCHED. 1997. *The PSCHED Initiative.*

http://parallel.nas.nasa.gov/Parallel/Projects/Psched/index.html

Sommers, Bret. 1997. "Agents: Not Just for Bond Anymore." *JavaWorld.* April.

http://www.javaworld.com/javaworld/jw-04-1997/jw-04-agents.html

Spring, Neil, and Rich Wolski. 1998. "Application Level Scheduling of Gene Comparison on Metacomputers." *Proceedings of the 12th ACM International Conference on Supercomputing*. July, Melbourne, Australia.

http://www.cs.ucsd.edu/groups/hpcl/apples/pubs/nspring-complib.ps

Sterling, Paul, Ian Foster, Carl Kesselman, Craig Lee, and Gregor von Laszewski. 1998. "A Fault Detection Service for Wide Area Distributed Computations." *Proceedings of the 7th Symposium on High-Performance Distributed Computing*. July, Chicago, IL.

ftp://ftp.globus.org/pub/globus/papers/hbm.pdf

Swarup, Vipin. 1997. "Trust Appraisal and Secure Routing of Mobile Agents." Position paper presented at the *DARPA Workshop on Foundations for Secure Mobile Code*, March 26-28, Monterey, CA.

http://www.cs.nps.navy.mil/research/languages/statements/swarup.ps

Vanhelsuwe, Laurence. 1997. "Create Your Own Supercomputer with Java." *JavaWorld.* January.

http://www.javaworld.com/javaworld/jw-01-1997/jw-01-dampp.html

Wolski, Rich. 1997. "Forecasting Network Performance to Support Dynamic Scheduling Using the Network Weather Service." *Proceedings of the 6th Symposium on High-Performance Distributed Computing*. August, Portland, OR, pp. 388-396.

ftp://ftp.cs.ucsd.edu/pub/rich/papers/nws-hpdc.ps.gz

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 1998	Final

4. TITLE AND SUBTITLE

WIDE-AREA, HETEROGENEOUS, DISTRIBUTED COMPUTING: TOWARD COMPUTATIONAL GRIDS

5. FUNDING NUMBERS

PE: 0603011F
AN: DN302112

6. AUTHOR(S)

C. V. Tran

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

SSC San Diego
53560 Hull Street
San Diego, CA 92152–5001

8. PERFORMING ORGANIZATION REPORT NUMBER

TD 3051

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

SAF/FMBMB
P. O. Box 46335
Washington, DC 20050–6395

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This document provides an overview of recent developments in several important aspects of wide-area, heterogeneous, distributed computing, including information on affordable, high-performance, commodity-computing clusters, a building block of computational grids. Workload-management software (WMS), another building block, is also discussed, and leading WMS packages are identified and described. Application of agent-based programming models to computational grid development is also covered.

14. SUBJECT TERMS

Mission Area: Computational Technology
computational grid
mobile agents
workload-management software
distributed computing

15. NUMBER OF PAGES

52

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

21a. NAME OF RESPONSIBLE INDIVIDUAL	21b. TELEPHONE *(include Area Code)*	21c. OFFICE SYMBOL
C. V. Tran	(619) 553–2524 e–mail:trancv@spawar.navy.mil	Code D73A

INITIAL DISTRIBUTION

Code D0012	Patent Counsel	(1)
Code D0271	Archive/Stock	(6)
Code D0274	Library	(2)
Code D027	M. E. Cathcart	(1)
Code D0271	D. Richter	(1)
Code D7213	J. Small	(1)
Code D73	T. Knight	(1)
Code D73A	S. I. Chou	(1)
Code D73A	C. Funk	(1)
Code D73A	D. Matsubara	(1)
Code D73A	D. Reese	(15)
Code D73A	C. Tran	(5)
Code D7305	K. Bromley	(1)

Defense Technical Information Center
Fort Belvoir, VA 22060–6218 (4)

SPAWARSYSCEN Liaison Office
Arlington, VA 22202–4804

Center for Naval Analyses
Alexandria, VA 22302–0268

Navy Acquisition, Research and Development
Information Center (NARDIC)
Arlington, VA 22244–5114

GIDEP Operations Center
Corona, CA 91718–8000